Project No. 10-886

Development of an Advanced Computational Fluid Dynamics Technology for the Next-Generation Nuclear Reactor System Analysis and Safety Margin Characterization Code

Reactor Concepts

Dr. Hong Luo
North Carolina State University

In collaboration with:
Idaho National Laboratory

Rich Reister, Federal POC
Robert Nourgaliev, Technical POC

NEUP Nuclear Energy University Programs
U.S. Department of Energy

Development of an Advanced Computational Fluid Dynamics Technology for the Next-Generation Nuclear Reactor System Analysis and Safety Margin Characterization Code

Project Number: 10-886
Final Report
10/15/2014

Submitted to
Dr. Val T. Seeley
NE University Program
CAES 272
Idaho National Laboratory
Idaho Falls, ID 83415-3840

Prepared by
Dr. Hong Luo
Department of Mechanical and Aerospace Engineering
North Carolina State University
911 Oval Dr. - EBIII 3236
Campus Box 7910
Raleigh, NC 27695-7910
e-mail: hong_luo@ncsu.edu

1. Summary

This report describes the research activities we have conducted at NCSU for our NEUP project. The work toward achieving the objectives of the project is reported. The significant achievements and accomplishments are presented. A number of numerical experiments are conducted to demonstrate that the goal of the proposed work has been successfully achieved. Issues, recommendations, and future work are discussed.

2. Introduction

The accuracy of many finite-volume and finite-element methods currently used in computational fluid dynamics is at best second order. These well established and fairly mature research and production CFD methods are able to provide orders of improvements in comparison to 1st order methods. In spite of an exhaustive effort on all possible refinements and improvements on their efficiency and robustness, the second-order CFD methods can hardly achieve the designed second-order accuracy in practice on unstructured tetrahedral grids, and therefore cannot deliver engineering-required accuracy in time for a variety of applications. Furthermore, Uncertainty Quantification (UQ) for a requested simulation using a second-order method can only be provided by a higher-order (>2nd) method.

Over the last several years, it has become clear that orders of magnitude improvements in both accuracy and efficiency can be achieved by replacing second-order methods with higher-order methods in CFD. This recognition has opened previously unimaginable opportunities to tackle a variety of complex flow problems in science and engineering. However, the promise of these methods has remained, to a great extent, unrealized because of the several difficulties raised by the application of these methods to flow problems of practical interests.

As the leader of higher-order methods in CFD applications, the discontinuous Galerkin (DG) methods[1-28] have received many attentions recently. The discontinuous Galerkin methods (DGM) combine two advantageous features commonly associated with finite element and finite volume methods (FVM). As in classical finite element methods, accuracy is obtained by means of high-order polynomial approximation within an element rather than by wide stencils as in the case of FVM. The physics of wave propagation is, however, accounted for by solving the Riemann problems that arise from the discontinuous representation of the solution at element interfaces. In this respect, the methods are therefore similar to FVM. What is known so far about these methods offers a tantalizing glimpse of their full potential. Indeed, what set these methods apart from the crowd are many distinct, attractive features they possess: 1) They have several useful mathematical properties with respect to conservation, stability, and convergence; 2) The methods can be easily extended for higher-order (>2nd) approximation; 3) The methods are well suited for complex geometries since they can be applied on arbitrary grids. In addition, the methods can also handle non-conforming elements, where the grids are allowed to have hanging nodes; 4) The methods are highly parallelizable, as they are compact and each element is independent. Since the elements are discontinuous, and the inter-element communications are minimal, domain decomposition can be efficiently employed. The compactness also allows for structured and simplified coding for the methods; 5) They can easily handle adaptive strategies, since refining or coarsening a grid can be achieved without considering the continuity restriction

commonly associated with the conforming elements. The methods allow easy implementation of *hp*-refinement, for example, the order of accuracy, or shape, can vary from element to element. 6) They have the ability to compute low Mach number flow problems without recourse to the time-preconditioning techniques normally required for the finite volume methods. In contrast to the enormous advances in the theoretical and numerical analysis of the DGM, the development of a viable, attractive, competitive, and ultimately superior DG method over the more mature and well-established second order methods is relatively an untouched area. This is mainly due to the fact that the DGM have a number of weaknesses that have yet to be addressed, before they can be robustly used for flow problems of practical interest in a complex configuration environment. In particular, there are three most challenging and unresolved issues in the DGM: a) how to efficiently discretize diffusion terms required for the Navier-Stokes equations, b) how to effectively control spurious oscillations in the presence of strong discontinuities, and c) how to develop efficient time integration schemes for time accurate and steady-state solutions. Indeed, compared to the finite element methods and finite volume methods, the DG methods require solutions of systems of equations with more unknowns for the same grids. Consequently, these methods have been recognized as expensive in terms of both computational costs and storage requirements.

Our research effort has been to bridge this gap between potential and reality by developing a higher-order reconstructed discontinuous Galerkin (rDG) method[18-28] that can provide significant improvements in accuracy and efficiency for solving a variety of complex flow problems compared to today's state-of-the-art second order methods. In reconstructed DG methods, termed rDG(PnPm), Pn indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and Pm represents a reconstructed polynomial solution of degree of m ($m \geq n$) that is used to compute the fluxes. The rDG(PnPm) schemes are designed to enhance the accuracy of the discontinuous Galerkin method by increasing the order of the underlying polynomial solution. The beauty of the rDG(PnPm) schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of rDG(PnPm) schemes, and thus allow for a direct efficiency comparison. When n=0, i.e. a piecewise constant polynomial is used to represent a numerical solution, rDG(P0Pm) is nothing but classical high order finite volume schemes, where a polynomial solution of degree m ($m \geq 1$) is reconstructed from a piecewise constant solution. When m=n, the reconstruction reduces to the identity operator, and rDG(PnPn) scheme yields a standard DG method. Our lately developed reconstructed discontinuous Galerkin method based on a hierarchical WENO reconstruction[36,37] is designed not only to reduce the high computing costs for the DG methods, but also to avoid spurious oscillations in the vicinity of strong discontinuities, thus effectively overcoming the two shortcomings of the DG methods. Our numerical experiments for a variety of flow problems indicate that the rDG(P1P2) method is able to capture shock waves within one cell without any spurious oscillations, achieve the designed third-order of accuracy: one order accuracy higher than the underlying DG method, and thus significantly increase its accuracy without significant increase in computing costs and memory requirements.

3. Research Activities

The main objective of the research effort in this project is to develop, apply, and

implement an advanced Computational Fluid Dynamics (CFD) technology that can be used in a next generation simulation code for design and safety analysis of advanced nuclear energy systems. The main research efforts involved in this project can be divided into three tasks: 1) Development and assessment of a third order spatial discretization method based on a reconstructed discontinuous Galerkin for the compressible Navier-Stokes equations on unstructured hybrid grids; 2) Development and implementation of a third-order implicit temporal discretization method for the rDG ; and 3) Verification and Validation of. The work performed in these three areas for this project is detailed below.

3.1 Development of a class of reconstructed DG methods on arbitrary grids

The objective of this task is to develop and assess a class of reconstructed discontinuous Galerkin methods for solving compressible flow problems on arbitrary grids. The reconstructed DG (rDG) methods, termed PnPm schemes and were introduced by Dumber et al.[18-20], where Pn indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and Pm represents a reconstructed polynomial solution of degree of m ($m \geq n$) that is used to compute the fluxes. The beauty of PnPm schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of PnPm schemes, and thus allow for a direct efficiency comparison. When n=0, i.e. a piecewise constant polynomial is used to represent a numerical solution, P0Pm is nothing but classical high order finite volume schemes, where a polynomial solution of degree m ($m \geq 1$) is reconstructed from a piecewise constant solution. When m=n, the reconstruction reduces to the identity operator, and PnPn scheme yields a standard DG method.

Obviously, the construction of an accurate and efficient reconstruction operator is crucial to the success of the PnPm schemes. In Dumbser's work, this is achieved using a so-called in-cell recovery, where recovered equations are obtained using a L2 projection, i.e., the recovered polynomial solution is uniquely determined by making it indistinguishable from the underlying DG solutions in the contributing cells in the weak sense. The resultant over-determined system is then solved using a least-squares method that guarantees exact conservation, not only of the cell averages but also of all higher order moments in the reconstructed cell itself, such as slopes and curvatures. However, this conservative least-squares recovery approach is computationally expensive, as it involves both recovery of a polynomial solution of higher order and least-squares solution of the resulting over-determined system. Furthermore, the recovery might be problematic for a boundary cell, where the number of the face-neighboring cells might be not enough to provide the necessary information to recover a polynomial solution of a desired order.

Fortunately, recovery is not the only way to obtain a polynomial solution of higher order from the underlying discontinuous Galerkin solutions. Rather, reconstruction widely used in the finite volume methods provides an alternative, probably a better choice to obtain a higher-order polynomial representation. Luo et al.[26-28] develop a reconstructed discontinuous Galerkin method using a Taylor basis[13] for the solution of the compressible Euler and Navier-Stokes equations on arbitrary grids, where a higher order polynomial solution is reconstructed by use of a strong interpolation, requiring point values and derivatives to be interpolated on the face-neighboring cells. The resulting over-determined linear system of equations is then solved in the least-squares sense. This reconstruction scheme only involves the von Neumann neighborhood, and thus is compact, simple, robust, and flexible. Furthermore, the reconstruction scheme guarantees exact

conservation, not only of the cell averages but also of their slopes due to a judicious choice of our Taylor basis.

More recently, Zhang et al.[29,30] presented a class of hybrid DG/FV methods for the conservation laws, where the second derivatives in a cell are obtained from the first derivatives in the cell itself and its neighboring cells using a Green-Gauss reconstruction widely used in the finite volume methods. This provides a fast, simple, and robust way to obtain a higher-order polynomial solutions. The numerical experiments indicate that this efficient reconstruction scheme is able to achieve a third-order accuracy: one order accuracy higher than the underlying second order DG method.

A comparative study has been conducted to assess the performance of these three reconstruction methods[31-32]. The numerical experiments indicate that all three reconstructed discontinuous Galerkin methods can deliver the desired third order of accuracy and significantly improve the accuracy of the underlying second-order DG method, although the least-squares reconstruction method provides the best performance in terms of both accuracy and robustness.

Unfortunately, the attempt to extend this rDG method to solve 3D Euler equations on tetrahedral grids was not successful. Like the second order cell-centered finite volume methods rDG(P0P1)[33], the resultant rDG(P1P2) method is unstable. Although rDG(P0P1) methods are in general stable in 2D and on Cartesian or structured grids in 3D, they suffer from the so-called linear instability on unstructured tetrahedral grids, when the reconstruction stencils only involve von Neumann neighborhood, i.e., adjacent face-neighboring cells[33]. The rDG(P1P2) method exhibits the same linear instability, which can be overcome by using extended stencils. However, this is achieved at the expense of sacrificing the compactness of the underlying DG methods. Furthermore, these linear reconstruction-based DG methods will suffer from non-physical oscillations in the vicinity of strong discontinuities for the compressible Euler equations. Alternatively, ENO, WENO, and HWENO can be used to reconstruct a higher-order polynomial solution, thereby not only enhancing the order of accuracy of the underlying DG method but also achieving both linear and non-linear stability. This type of hybrid HWENO+DG schemes has been presented on 1D and 2D structured grids by Balsara et al.[34], where the HWENO reconstruction is relatively simple and straightforward.

Our effort has been focused on developing a Reconstructed Discontinuous Galerkin method, rDG(P1P2), based on a WENO reconstruction using a Taylor basis[13] for solving compressible flow problems on hybrid grids. This rDG(P1P2) method is designed not only to reduce the high computing costs of the DGM, but also to avoid spurious oscillations in the vicinity of strong discontinuities, thus effectively addressing the two shortcomings of the DGM. In this rDG(P1P2) method, a quadratic solution is first reconstructed to enhance the accuracy of the underlying DG method in two steps: (1) all second derivatives on each cell are first reconstructed using the solution variables and their first derivatives from adjacent face-neighboring cells via a strong interpolation; (2) the final second derivatives on each cell are then obtained using a WENO strategy based on the reconstructed second derivatives on the cell itself and its adjacent face-neighboring cells. This reconstruction scheme, by taking advantage of handily available and yet valuable information namely the gradients in the context of the DG methods, only involves von Neumann neighborhood and thus is compact, simple, robust, and

flexible. As the underlying DG method is second-order, and the basis functions are at most linear functions, fewer quadrature points are then required for both domain and face integrals, and the number of unknowns (the number of degrees of freedom) remains the same as for the DG(P1). Consequently, this rDG method is more efficient than its third order DG(P2) counterpart. The gradients of the quadratic polynomial solutions are then modified using a WENO reconstruction in order to eliminate non-physical oscillations in the vicinity of strong discontinuities, thus ensuring the non-linear stability of the RDG method. The developed rDG(P1P2) method is used to compute a variety of flow problems ranging from nearly incompressible flows to supersonic flows on hybrid grids to demonstrate its accuracy, robustness, versatility, and essentially non-oscillatory property. The presented numerical results indicate that this rDG(P1P2) method is able to 1) capture shock waves sharply essentially without any spurious oscillations, 2) provide the accurate simulations for a wide range of flow regimes from nearly incompressible flows to supersonic flows without using time-derivatives preconditioning methods, without modifying the Riemann flux functions, and without adjusting any parameters, and 3) achieve the designed third-order of accuracy for smooth flows: one order accuracy higher than the underlying DG(P1) method, and thus significantly increase its accuracy without significant increase in computing costs and memory requirements.

Our work on the development of this rDG method has been published in a number of papers. Two papers published in journal of computational physics are attached in appendix 1 of this report for the sake of completeness.

3.2. Development of an accurate, efficient higher-order method for temporal discretization

Our ultimate objective is to develop an overall third-order accurate numerical method in both space and time for the solution of the compressible Navier-Stokes equations on 3D hybrid grids. The focus of this task is placed on the development of a robust high-order fully-implicit temporal discretization for the rDG methods. This research work is strongly motivated by the need to develop an accurate and fast arbitrary high-order implicit method for solving time-accurate flow problems in order to keep the overall higher-accuracy of the higher-order reconstructed discontinuous Galerkin methods. It has been conclusively demonstrated in the literature that the use of higher order methods in space alone does not ensure a more accurate solution in that the error deduced by the time-stepping methods can be dominant. Explicit methods such as multi-stage Runge-Kutta schemes may be the only choice for certain unsteady applications such as shock wave and transition simulations, when the time scales of interest are small, or more precisely, when they are comparable to the spatial scales. However, when dealing with many low reduced frequency phenomena with disparate temporal and spatial scales, explicit methods are notoriously time-consuming, since the allowable time step is much more restrictive than that needed for an acceptable level of time accuracy. Therefore, it is desirable to develop a fully implicit method, where the time step is solely determined by the temporal accuracy consideration for the flow physics and is not limited by the numerical stability consideration. Implicit methods, such as the first-order accurate backward Euler scheme, the Crank-Nicholson method, and the second-order backward differentiation formula, can be used for these types of problems. These time integration schemes are relatively efficient because they solve only one implicit set of equations per time step. However, they require a fixed time step, thus rendering them less efficient. They are not A-stable, thus rendering them less robust. They only provide a second-order temporal accuracy, thus rendering them less accurate. The development of accurate

and fast arbitrary high-order implicit methods is needed in order to keep the overall higher-accuracy and higher-efficiency of the higher-order discontinuous Galerkin methods. Recently, a diagonally implicit Runge-Kutta (IRK) method, originally developed by Bijl et al[38] for the finite volume solutions of the Navier-Stokes equations is extended by Wang et al.[39] to solve the compressible Euler equations using higher-order discontinuous Galerkin methods. They conclude that the diagonally implicit Runge-Kutta method is more efficient than the second-order time integration schemes. We have extended and implemented IRK method for the time accurate solutions of the compressible Navier-Stokes equations on arbitrary grids using the reconstructed discontinuous Galerkin method[40,41]. A system of nonlinear equations arising from a diagonally implicit Runge-Kutta temporal discretization of the unsteady Euler and Navier-Stokes equations is solved at each time step using a pseudo-time marching approach. The resulting systems of linear algebraic equations are solved using the GMRES+LU-SGS method. Three approaches: analytical derivation, divided differencing, and automatic differentiation (AD) are presented, developed, and compared to construct the Jacobian matrix. The developed implicit rDG method is used to compute a variety of unsteady flow problems on 3D hybrid grids. The numerical results obtained indicate that the use of the present implicit methods lead to orders of improvements in performance over its explicit counterpart, while without significant increase in memory requirements and that the implicit method where the construction of Jacobian matrix is based on the AD approach performs the best in terms of robustness and efficiency.

Our work on the development of implicit methods for the rDG method has been published in a number of papers. Two papers published in journal of Computers & Fluids are attached in appendix 2 of this report for the sake of completeness.

4. Conclusions and Recommendations

A reconstructed discontinuous Galerkin method has been developed for the compressible flows at all speeds. The developed rDG has been used to compute a variety of flow problems to assess its accuracy and test its robustness for a variety of compressible flows. The numerical experiments clearly demonstrate that the developed rDG(P_1P_2) method is able to achieve the designed third-order accuracy: one order accuracy higher than the underlying DG method, and obtain the accurate solutions for a wide range of flow regimes from nearly incompressible flows to supersonic flows without adjustment of any parameters and without recourse to the time-derivative preconditioning methods.

Although the execution of this project meets or exceeds our initial expectations, there are so many topics that can be pursued as a follow-up of this project. In particular, 1) implementation of this rDG method in RELAP-7 for hydraulics in the framework of MOOSE and 2) development of a two-phase flow capability using the rDG method in the framework of MOOSE, are two projects on which we are ready to collaborate with our colleagues at INL.

5. Accomplishments

The most significant accomplishment in this project is probably that we are able to demonstrate the accuracy, efficiency, robustness, and non-oscillatory property of our rDG

method. The developed rDG(P1P2) method not only enhances the accuracy of discontinuous Galerkin method but also avoids spurious oscillation in the vicinity of discontinuities, which is crucial for the simulation of the two phase flow problems. This rDG method truly signifies a giant leap towards development of higher-order CFD code for application of complex flow problems in nuclear engineering.

References

1. W.H. Reed and T.R. Hill, ``Triangular Mesh Methods for the Neutron Transport Equation," **Los Alamos Scientific Laboratory Report**, LA-UR-73-479, 1973.
2. B. Cockburn, S. Hou, and C. W. Shu, ``TVD Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for conservation laws IV: the Multidimensional Case," **Mathematics of Computation**, Vol. 55, pp. 545-581, 1990.eorge, P. L., Automatic Mesh Generation, J. Wiley & Sons, 1991.
3. B. Cockburn, and C. W. Shu, ``The Runge-Kutta Discontinuous Galerkin Method for conservation laws V: Multidimensional System," **Journal of Computational Physics**, Vol. 141, pp. 199-224, 1998.
4. B. Cockburn, G. Karniadakis, and C. W. Shu, ``The Development of Discontinuous Galerkin Method", in Discontinuous Galerkin Methods, Theory, Computation, and Applications, edited by B. Cockburn, G.E. Karniadakis, and C. W. Shu, Lecture Notes in Computational Science and Engineering, Springer-Verlag, New York, 2000, Vol. 11 pp. 5-50, 2000.
5. F. Bassi and S. Rebay, ``High-Order Accurate Discontinuous Finite Element Solution of the 2D Euler Equations," **Journal of Computational Physics**, Vol. 138, pp. 251-285, 1997.
6. H. L. Atkins and C. W. Shu, ``Quadrature Free Implementation of Discontinuous Galerkin Method for Hyperbolic Equations," **AIAA Journal**, Vol. 36, No. 5, 1998.
7. F. Bassi and S. Rebay, ``GMRES discontinuous Galerkin solution of the Compressible Navier-Stokes Equations," **Discontinuous Galerkin Methods, Theory, Computation, and Applications**, edited by B. Cockburn, G.E. Karniadakis, and C. W. Shu, Lecture Notes in Computational Science and Engineering, Springer-Verlag, New York, 2000, Vol. 11 pp. 197-208, 2000.
8. T. C. Warburton, and G. E. Karniadakis, ``A Discontinuous Galerkin Method for the Viscous MHD Equations," **Journal of Computational Physics**, Vol. 152, pp. 608-641, 1999.
9. J. S. Hesthaven and T. Warburton, ``Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications," Texts in **Applied Mathematics**, Vol. 56, 2008.
10. P. Rasetarinera and M. Y. Hussaini, ``An Efficient Implicit Discontinuous Spectral Galerkin Method," **Journal of Computational Physics**, Vol. 172, pp. 718-738, 2001.
11. B. T. Helenbrook, D. Mavriplis, and H. L. Atkins, ``Analysis of *p*-Multigrid for Continuous and Discontinuous Finite Element Discretizations," AIAA Paper 2003-3989, 2003.
12. K. J. Fidkowski, T. A. Oliver, J. Lu, and D. L. Darmofal, ``*p*-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations," **Journal of Computational Physics**, Vol. 207, No. 1, pp. 92-113, 2005.
13. H. Luo, J. D. Baum, and R. Löhner, ``A Discontinuous Galerkin Method Using Taylor Basis for Compressible Flows on Arbitrary Grids," **Journal of Computational Physics**, Vol. 227, No 20, pp. 8875-8893, October 2008.

14. H. Luo, J.D. Baum, and R. Löhner, ``On the Computation of Steady-State Compressible Flows Using a Discontinuous Galerkin Method", **International Journal for Numerical Methods in Engineering,** Vol. 73, No. 5, pp. 597-623, 2008.
15. H. Luo, J. D. Baum, and R. Löhner, ``A Hermite WENO-based Limiter for Discontinuous Galerkin Method on Unstructured Grids," **Journal of Computational Physics**, Vol. 225, No. 1, pp. 686-713, 2007.
16. H. Luo, J.D. Baum, and R. Löhner, ``A *p*-Multigrid Discontinuous Galerkin Method for the Euler Equations on Unstructured Grids", **Journal of Computational Physics**, Vol. 211, No. 2, pp. 767-783, 2006.
17. H. Luo, J.D. Baum, and R. Löhner, ``Fast, *p*-Multigrid Discontinuous Galerkin Method for Compressible Flows at All Speeds", **AIAA Journal** , Vol. 46, No. 3, pp.635-652, 2008.
18. M. Dumbser, D.S. Balsara, E.F. Toro, C.D. Munz. A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes. **Journal of Computational Physics**, 227:8209-8253, 2008.
19. M. Dumbser, O. Zanotti. Very high order PNPM schemes on unstructured meshes for the resistive relativistic MHD equations. **Journal of Computational Physics**, 228:6991-7006, 2009.
20. M. Dumbser. Arbitrary High Order PNPM Schemes on Unstructured Meshes for the Compressible Navier-Stokes Equations. **Computers & Fluids**, 39: 60-76. 2010.
21. F. Bassi and S. Rebay, ``A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier-Stokes Equations," **Journal of Computational Physics**, Vol. 131, pp. 267-279, 1997.
22. F. Bassi and S. Rebay, ``Discontinuous Galerkin Solution of the Reynolds-Averaged Navier-Stokes and k-ω Turbulence Model Equations, **Journal of Computational Physics**, Vol. 34, pp. 507-540, 2005.
23. B. Cockburn and C.W. Shu, ``The Local Discontinuous Galerkin Method for Time-dependent Convection-Diffusion System, **SIAM, Journal of Numerical Analysis**, Vo. 16, 2001.
24. C. E. Baumann and J. T. Oden, ``A Discontinuous *hp* Finite Element Method for the Euler and Navier-Stokes Equations," **International Journal for Numerical Methods in Fluids**, Vol. 31, 1999.
25. H. Luo, L. Luo, and K. Xu - A Discontinuous Galerkin Method Based on a BGK Scheme for the Navier-Stokes Equations on Arbitrary Grids, **Advances in Applied Mathematics and Mechanics**, Vol. 1, No. 3, pp. 301-318, 2009.
26. H. Luo, L. Luo, and R. Nourgaliev, A Reconstructed Discontinuous Galerkin Method for the Euler Equations on Arbitrary Grids, **Communication in Computational Physics,** Vol. 12, No. 5, pp. 1495-1519, 2012.
27. H. Luo, L. Luo, R. Nourgaliev, V.A. Mousseau, and N. Dinh, ``A Reconstructed Discontinuous Galerkin Method for the Compressible Navier-Stokes Equations on Arbitrary Grids", **Journal of Computational Physics**, Vol. 229, pp. 6961-6978, 2010.
28. H. Luo, L. Luo, A. Ali, R. Nourgaliev, and C. Cai, ``A Parallel, Reconstructed Discontinuous Galerkin Method for the Compressible Flows on Arbitrary Grids", **Communication in Computational Physics**, Vol. 9, No. 2, pp. 363-389, 2011.
29. L.P. Zhang, W. Liu, L.X. He, X.G. Deng, and H.X. Zhang, A Class of Hybrid DG/FV Methods for Conservation Laws I: Basic Formulation and One-Dimensional System, **Journal of Computational Physics,** Vol. 23, No. 4, pp. 1081-1103, 2012.

30. L.P. Zhang, W. Liu, L.X. He, X.G. Deng, and H.X. Zhang, A Class of Hybrid DG/FV Methods for Conservation Laws II: Two dimensional Cases, **Journal of Computational Physics,** Vol. 23, No. 4, pp. 1104-1120, 2012.
31. H. Luo, Y. Xia, R. Nourgaliev, and C. Cai, A Class of Reconstructed discontinuous Galerkin Methods for the Compressible Flows on Arbitrary Grids, AIAA-2011-0199, 2011.
32. H. Luo, H. Xiao, R. Nourgaliev, and C. Cai, A Comparative Study of Different Reconstruction Schemes for Reconstructed Discontinuous Galerkin Methods for the Compressible Flows on Arbitrary Grids, AIAA-2011-3839, 2011.
33. D. F. Haider, J.P. Croisille, and B. Courbet, Stability Analysis of the Cell Centered Finite-Volume MUSCL Method on Unstructured Grids, **Numirische Mathematik**, Vol. 113, No. 4 pp. 555-600, 2009.
34. D. Balsara, C. Altmann, C.D. Munz and M. Dumbser, A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG + HWENO schemes, **Journal of Computational Physics**, Vol. **226**, pp. 586–620, 2007.
35. Z. Xu, Y Liu, H Du, and CW Shu, Point-wise hierarchical Reconstruction for Discontinuous Galerkin and Finite Volume Method for Solving Conservation Laws, **Journal of Computational Physics**, Vol. **230**, pp. 6843-6865, 2011.
36. H. Luo, H., Y. Xia, S. Li, R. Nourgaliev, and C. Cai, A Hermite WENO Reconstruction-Based Discontinuous Galerkin Method for the Euler Equations on Tetrahedral Grids, **Journal of Computational Physics**, Vol. 231, pp. 5489-5503, 2012.
37. Luo, H., Xia Y., Spiegel, S., Nourgaliev, R., Jiang, Z., A Reconstructed Discontinuous Galerkin Method Based on a Hierarchical WENO Reconstruction for Compressible Flows on Tetrahedral Grids, **Journal of Computational Physics**, Vol. 236, pp. 477-492, 2013.
38. H. Bijl, M.H. Carpenter, V.N. Vasta, and C. A. Kennedy, Implicit Time Integration Schemes for the Unsteady Compressible Navier-Stokes Equations: Laminar Flow, **Journal of Computational Physics**, Vol. 179, pp. 313-329, 2002.
39. L. Wang and D. Mavriplis, Implicit Solution of the Unsteady Euler Equations for High-Order Accurate Discontinuous Galerkin Discretization, **Journal of Computational Physics**, Vol. 255, No. 2, pp. 1994-2015, 2007.
40. Xia, Y., Luo, H., and Nourgaliev, R., An implicit Hermite WENO reconstruction-based discontinuous Galerkin method on tetrahedral grids, **Computers & Fluids**, Vol. 96, pp. 406-421, 2014.
41. Xia, Y., Luo, H., Frisbey, M., and Nourgaliev, R., A Set of Parallel, Implicit Methods for a Reconstructed Discontinuous Galerkin Method for the Compressible Flows on 3D Arbitrary Grids, **Computers & Fluids**, Vol. 98, pp. 134-151, 2014.

Publications

Referred Journal Articles

1. Xia, Y., Luo, H., and Nourgaliev, R., An implicit Hermite WENO reconstruction-based discontinuous Galerkin method on tetrahedral grids, **Computers & Fluids**, 2014, http://dx.doi.org/10.1016/j.compfluid.2014.02.027

2. Xia, Y., Luo, H., Frisbey, M., and Nourgaliev, R., A Set of Parallel, Implicit Methods for a Reconstructed Discontinuous Galerkin Method for the Compressible Flows on 3D Arbitrary Grids, **Computers & Fluids**, 2014, http://dx.doi.org/10.10.16/j.compfluid.2014.01.023

3. Luo, H., Xia Y., Spiegel, S., Nourgaliev, R., Jiang, Z., A Reconstructed Discontinuous Galerkin Method Based on a Hierarchical WENO Reconstruction for Compressible Flows on Tetrahedral Grids, **Journal of Computational Physics**, Vol. 236, pp. 477-492, 2013

4. Xia, Y., Luo, H., and Nourgaliev, R., Implicit Solution for a Hermite WENO Reconstruction-Based Discontinuous Galerkin Method on Tetrahedral Grids, **Theoretical and Applied Mechanics Letters**, Vol. 2, No. 042002, 2012.

5. Luo, H., Xia Y., Li. S., Nourgaliev, R., and Cai, C., A Hermite WENO Reconstruction-Based Discontinuous Galerkin Method for the Euler Equations on Tetrahedral Grids, **Journal of Computational Physics**, Vol. 231, pp. 5489-5503, 2012.

6. Luo, H., Luo, L., and Nourgaliev, R., A Reconstructed Discontinuous Galerkin Method for the Euler Equations on Arbitrary Grids, **Communication in Computational Physics**, Vol. 12, No. 5, pp. 1495-1519, 2012.

Refereed Conference Papers

1. Yidong Xia, Hong Luo, Chuanjin Wang, and Robert Nourgalev, An implicit, reconstructed discontinuous Galerkin method for the unsteady compressible Navier-Stokes equations on 3D hybrid grids, AIAA-2014-3220, 7th AIAA Theoretical Fluid Mechanics Conference, Atlanta, GA, June16-20, 2014

2. Yidong Xia, Hong Luo, Chuanjin Wang, and Robert Nourgalev, Implicit Large Eddy Simulation of Turbulent Flows Using a Reconstructed Discontinuous Galerkin Method, AIAA-2014-0224, 52nd Aerospace Sciences Meeting, National Harbor, MD, Jan. 13-17, 2014.

3. Yidong Xia, Hong Luo, Seth C. Spiegel, Megan Frisbey, and Robert Nourgaliev, A Parallel, Implicit Reconstructed Discontinuous Galerkin Method for the Compressible Flows on 3D Arbitrary Grids, AIAA-2013-3062, 21st AIAA Computational Fluid Dynamics Conference, San Diego, CA, 24-27 June, 2013.

4. Yidong Xia, Megan Frisbey, and Hong Luo, A Reconstructed Discontinuous Galerkin Method Based on a Hierarchical WENO Reconstruction for Computing Shock Waves on Hybrid Grids, AIAA-2013-3063, 21st AIAA Computational Fluid Dynamics Conference, San Diego, CA, 24-27 June, 2013.

5. Luo, H., Xia Y, Frisbey, F., and Nourgaliev, R. A WENO Reconstruction-Based Discontinuous Galerkin Method for Compressible Flows on Hybrid Grids, AIAA-2013-0516, 51st AIAA Aerospace Sciences Meeting, Grapevine, Texas, Jan. 7-10, 2013.

6. Xia Y, Luo, H., and Nourgaliev, R. An Implicit Reconstructed Discontinuous Galerkin Method Based on Automatic Differentiation for the Navier-Stokes Equations on Tetrahedron Grids, AIAA-2013-0687, 51st AIAA Aerospace Sciences Meeting, Grapevine, Texas, Jan. 7-10, 2013.

7. Xia Y, Luo H., and Nourgaliev, R., An Implicit Reconstructed Discontinuous Galerkin Method on Tetrahedral Grids, *Seventh International Conference on Computational Fluid Dynamics*, Hawaii, USA, July 9-13, 2012.

8. Luo, H., Xia Y., and Nourgaliev, R., A Hermit WENO Reconstruction-Based Discontinuous Galerkin Method for Compressible Flows on Tetrahedral Grids; *Seventh International Conference on Computational Fluid Dynamics*, Hawaii, USA, July 9-13, 2012.

9. Luo, H., Xia Y, Nourgaliev, R. A Hierarchical Hermit WENO Reconstruction-Based Discontinuous Galerkin Method for the Compressible Flows on Tetrahedral Grids, AIAA-2012-2838, 42nd AIAA Fluid Dynamics Conference and Exhibit, 25-28 June 2012, New Orleans, LA.

10. Xia Y, Luo H., and Nourgaliev, R., An Implicit Method for a Reconstructed Discontinuous Galerkin Method on Tetrahedral Grids, AIAA-2012-2834, 42nd AIAA Fluid Dynamics Conference and Exhibit, 25-28 June 2012, New Orleans, LA.

11. Luo, H., Li, SJ, Xia Y, Nourgaliev, R., and Cai C., A Hermit WENO Reconstruction-based Discontinuous Galerkin Method for the Euler Equations on Tetrahedral Grids, AIAA-2012-0461, 50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, Nashville, Tennessee, Jan. 9-12, 2012.

Appendix 1

Representative publications regarding the development of rDG methods

Journal of Computational Physics 231 (2012) 5489–5503

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A Hermite WENO reconstruction-based discontinuous Galerkin method for the Euler equations on tetrahedral grids

Hong Luo [a,*], Yidong Xia [a], Shujie Li [a], Robert Nourgaliev [b], Chunpei Cai [c]

[a] Department of Mechanical and Aerospace Engineering, North Carolina State University, Raleigh, NC 27695, United States
[b] Idaho National Laboratory, Idaho Falls, ID 83415, United States
[c] Department of Mechanical and Aerospace Engineering, New Mexico State University, Las Cruces, NM 88001, United States

ARTICLE INFO

Article history:
Received 23 December 2011
Received in revised form 6 April 2012
Accepted 4 May 2012
Available online 17 May 2012

Keywords:
Discontinuous Galerkin method
Hermite WENO reconstruction
Compressible Euler equations
Tetrahedral grids

ABSTRACT

A Hermite WENO reconstruction-based discontinuous Galerkin method RDG(P_1P_2), designed not only to enhance the accuracy of discontinuous Galerkin method but also to ensure linear stability of the RDG method, is presented for solving the compressible Euler equations on tetrahedral grids. In this RDG(P_1P_2) method, a quadratic polynomial solution (P_2) is first reconstructed using a least-squares method from the underlying linear polynomial (P_1) discontinuous Galerkin solution. By taking advantage of handily available and yet invaluable information, namely the derivatives in the DG formulation, the stencils used in the reconstruction involve only von Neumann neighborhood (adjacent face-neighboring cells) and thus are compact and consistent with the underlying DG method. The final quadratic polynomial solution is then obtained using a WENO reconstruction, which is necessary to ensure linear stability of the RDG method. The developed RDG method is used to compute a variety of flow problems on tetrahedral meshes to demonstrate its accuracy, efficiency, robustness, and versatility. The numerical experiments demonstrate that the developed RDG(P_1P_2) method is able to maintain the linear stability, achieve the designed third-order of accuracy: one order accuracy higher than the underlying DG method without significant increase in computing costs and storage requirements.

1. Introduction

The discontinuous Galerkin methods [1–28] (DGM) have recently become popular for the solution of systems of conservation laws. Originally introduced for the solution of neutron transport equations [1], nowadays they are widely used in computational fluid dynamics, computational acoustics, and computational magneto-hydrodynamics. The discontinuous Galerkin methods combine two advantageous features commonly associated with finite element and finite volume methods. As in classical finite element methods, accuracy is obtained by means of high-order polynomial approximation within an element rather than by wide stencils as in the case of finite volume methods. The physics of wave propagation is, however, accounted for by solving the Riemann problems that arise from the discontinuous representation of the solution at element interfaces. In this respect, the DG methods are similar to finite volume methods. The discontinuous Galerkin methods have many attractive features: (1) they have several useful mathematical properties with respect to conservation, stability, and convergence; (2) the methods can be easily extended to higher-order (>2nd) approximation; (3) the methods are well suited for complex geometries since they can be applied on unstructured grids. In addition, the methods can also handle non-conforming elements, where the grids are allowed to have hanging nodes; (4) the methods are highly parallelizable, as they are

* Corresponding author.
E-mail address: hong_luo@ncsu.edu (H. Luo).

0021-9991/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2012.05.011

compact and each element is independent. Since the elements are discontinuous, and the inter-element communications are minimal, domain decomposition can be efficiently employed. The compactness also allows for structured and simplified coding for the methods; (5) they can easily handle adaptive strategies, since refining or coarsening a grid can be achieved without considering the continuity restriction commonly associated with the conforming elements. The methods allow easy implementation of *hp*-refinement, for example, the order of accuracy, or shape, can vary from element to element; (6) they have the ability to compute low Mach number flow problems without recourse to the time-preconditioning techniques normally required for the finite volume methods. In contrast to the enormous advances in the theoretical and numerical analysis of the DGM, the development of a viable, attractive, competitive, and ultimately superior DG method over the more mature and well-established second order finite volume methods is relatively an untouched area. This is mainly due to the fact that the DGM have a number of weaknesses that have yet to be addressed, before they can be robustly used for flow problems of practical interest in a complex configuration environment. In particular, how to effectively control spurious oscillations in the presence of strong discontinuities, and how to reduce the computing costs for the DGM remain the two most challenging and unresolved issues in the DGM. Indeed, compared to the finite element methods and finite volume methods, the DGM require solutions of systems of equations with more unknowns for the same grids. Consequently, these methods have been recognized as expensive in terms of both computational costs and storage requirements.

In order to reduce high costs associated with the DGM, Dumbser et al. [18–20] have introduced a new family of reconstructed DGM, termed P_nP_m schemes and referred to as RDG(P_nP_m) in this paper, where P_n indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and P_m represents a reconstructed polynomial solution of degree of m ($m \geqslant n$) that is used to compute the fluxes. The RDG(P_nP_m) schemes are designed to enhance the accuracy of the discontinuous Galerkin method by increasing the order of the underlying polynomial solution. The beauty of RDG(P_nP_m) schemes is that they provide a unified formulation for both finite volume and DGM, and contain both classical finite volume and standard DG methods as two special cases of RDG(P_nP_m) schemes, and thus allow for a direct efficiency comparison. When $n = 0$, i.e. a piecewise constant polynomial is used to represent a numerical solution, RDG(P_0P_m) is nothing but classical high order finite volume schemes, where a polynomial solution of degree $m(m \geqslant 1)$ is reconstructed from a piecewise constant solution. When $m = n$, the reconstruction reduces to the identity operator, and RDG(P_nP_n) scheme yields a standard DG method.

Obviously, the construction of an accurate and efficient reconstruction operator is crucial to the success of the RDG(P_nP_m) schemes. In Dumbser's work, a higher order polynomial solution is reconstructed using a L_2 projection, requiring it indistinguishable from the underlying DG solutions in the contributing cells in the weak sense. The resultant over-determined system is then solved using a least-squares method that guarantees exact conservation, not only of the cell averages but also of all higher order moments in the reconstructed cell itself, such as slopes and curvatures. However, this conservative least-squares reconstruction approach is computationally expensive, as the L_2 projection, i.e., the operation of integration, is required to obtain the resulting over-determined system. Furthermore, the reconstruction might be problematic for a boundary cell, where the number of the face-neighboring cells might be not enough to provide the necessary information to recover a polynomial solution of a desired order. However, the projection-based reconstruction is not the only way to obtain a polynomial solution of higher order from the underlying discontinuous Galerkin solutions. In a reconstructed DG method using a Taylor basis [26–28] developed by Luo et al. for the solution of the compressible Euler and Navier–Stokes equations on arbitrary grids, a higher order polynomial solution is reconstructed by use of a strong interpolation, requiring point values and derivatives to be interpolated on the face-neighboring cells. The resulting over-determined linear system of equations is then solved in the least-squares sense. This reconstruction scheme only involves von Neumann neighborhood, and thus is compact, simple, robust, and flexible. Like the projection-based reconstruction, the strong reconstruction scheme guarantees exact conservation, not only of the cell averages but also of their slopes due to a judicious choice of the Taylor basis. More recently, Zhang et al. [29,30] presented a class of hybrid DG/FV methods for the conservation laws, where the second derivatives in a cell are obtained from the first derivatives in the cell itself and its neighboring cells using a Green–Gauss reconstruction widely used in the finite volume methods. This provides a fast, simple, and robust way to obtain higher-order polynomial solutions. Lately, Luo et al. [31,32] have conducted a comparative study for these three reconstructed discontinuous Galerkin methods RDG(P_1P_2) by solving 2D Euler equations on arbitrary grids. It is found that all three reconstructed discontinuous Galerkin methods can deliver the desired third order of accuracy and significantly improve the accuracy of the underlying second-order DG method, although the least-squares reconstruction method provides the best performance in terms of both accuracy and robustness.

However, the attempt to extend our RDG method to solve 3D Euler equations on tetrahedral grids was not successful. Like the second order cell-centered finite volume methods, i.e., RDG(P_0P_1), the resultant RDG(P_1P_2) methods are unstable. Although RDG(P_0P_1) methods are in general stable in 2D and on Cartesian or structured grids in 3D, they suffer from the so-called *linear instability* on unstructured tetrahedral grids, when the reconstruction stencils only involve von Neumann neighborhood, i.e., adjacent face-neighboring cells [33]. Unfortunately, the RDG(P_1P_2) method exhibits the same linear instability, which can be overcome by using extended stencils. However, this is achieved at the expense of sacrificing the compactness of the underlying DG methods. Furthermore, these linear reconstruction-based DG methods will suffer from non-physical oscillations in the vicinity of strong discontinuities for the compressible Euler equations. Alternatively, ENO, WENO, and HWENO can be used to reconstruct a higher-order polynomial solution, thereby not only enhancing the order of accuracy of the underlying DG method but also achieving both linear and non-linear stability. This type of hybrid HWENO+DG schemes has been developed on 1D and 2D structured grids by Balsara et al. [34], where the HWENO reconstruction is relatively simple and straightforward.

The objective of the effort discussed in this paper is to develop a Reconstructed Discontinuous Galerkin method, RDG(P_1P_2), based on a Hermite WENO reconstruction using a Taylor basis [13] for the solution of the compressible Euler equations on unstructured tetrahedral grids. This HWENO-based RDG method is designed not only to reduce the high computing costs of the DGM, but also to avoid spurious oscillations in the vicinity of strong discontinuities, thus effectively overcoming the two shortcomings of the DGM and ensuring the stability of the reconstructed DG method. In this RDG(P_1P_2) method, a quadratic solution is obtained in two steps: (1) all second derivatives on each cell are first reconstructed using the solution variables and their first derivatives from adjacent face-neighboring cells via a strong interpolation; (2) the final second derivatives on each cell are then obtained using a WENO strategy based on the reconstructed second derivatives on the cell itself and its adjacent face-neighboring cells. This reconstruction scheme, by taking advantage of handily available and yet valuable information namely the gradients in the context of the DG methods, only involves von Neumann neighborhood and thus is compact, simple, robust, and flexible. As the underlying DG method is second-order, and the basis functions are at most linear functions, fewer quadrature points are then required for both domain and face integrals, and the number of unknowns (the number of degrees of freedom) remains the same as for the DG(P_1). Consequently, this RDG method is more efficient than its third order RDG(P_2P_2) counterpart. The developed RDG method is used to compute a variety of flow problems on tetrahedral grids to demonstrate its accuracy, efficiency, and robustness. The presented numerical results indicate that this Hermite WENO reconstruction-based RDG(P_1P_2) method is able to achieve the designed third-order of accuracy: one order accuracy higher than the underlying DG(P_1) method, and thus significantly increase its accuracy without significant increase in computing costs and memory requirements. The remainder of this paper is organized as follows. The governing equations are listed in Section 2. The underlying reconstructed discontinuous Galerkin method is presented in Section 3. Extensive numerical experiments are reported in Section 4. Concluding remarks are given in Section 5.

2. Governing equations

The Euler equations governing unsteady compressible inviscid flows can be expressed as

$$\frac{\partial \mathbf{U}(x,t)}{\partial t}+\frac{\partial \mathbf{F}_k(\mathbf{U}(x,t))}{\partial x_k}=0 \tag{2.1}$$

where the summation convention has been used. The conservative variable vector **U**, and inviscid flux vector **F** are defined by

$$\mathbf{U}=\begin{pmatrix}\rho\\ \rho u_i\\ \rho e\end{pmatrix}\quad \mathbf{F}_j=\begin{pmatrix}\rho u_j\\ \rho u_i u_j+p\delta_{ij}\\ u_j(\rho e+p)\end{pmatrix} \tag{2.2}$$

Here ρ, p, and e denote the density, pressure, and specific total energy of the fluid, respectively, and u_i is the velocity of the flow in the coordinate direction x_i. The pressure can be computed from the equation of state

$$p=(\gamma-1)\rho\left(e-\frac{1}{2}u_j u_j\right) \tag{2.3}$$

which is valid for perfect gas, where γ is the ratio of the specific heats.

3. Reconstructed discontinuous Galerkin method

3.1. Discontinuous Galerkin formulation

The governing equation (2.1) is discretized using a discontinuous Galerkin finite element formulation. To formulate the discontinuous Galerkin method, we first introduce the following weak formulation, which is obtained by multiplying the above conservation law by a test function W, integrating over the domain Ω, and then performing an integration by parts,

$$\int_\Omega \frac{\partial \mathbf{U}}{\partial t}\mathbf{W}d\Omega+\int_\Gamma \mathbf{F}_k\mathbf{n}_k\mathbf{W}d\Gamma-\int_\Omega \mathbf{F}_k\frac{\partial \mathbf{W}}{\partial x_k}d\Omega=0,\quad \forall \mathbf{W}\in V \tag{3.1}$$

where $\Gamma(=\partial\Omega)$ denotes the boundary of Ω, and $\mathbf{n}_j$ the unit outward normal vector to the boundary. We assume that the domain Ω is subdivided into a collection of non-overlapping tetrahedral elements Ω_e. We introduce the following broken Sobolev space V_h^p

$$V_h^p=\{v_h\in[L_2(\Omega)]^m: v_h|_{\Omega_e}\in[V_p^m]\forall\Omega_e\in\Omega\} \tag{3.2}$$

which consists of discontinuous vector-values polynomial functions of degree p, and where m is the dimension of the unknown vector and

$$V_p^m=\text{span}\left\{\prod x_i^{\alpha_i}: 0\leqslant\alpha_i\leqslant p,\ 0\leqslant i\leqslant d\right\} \tag{3.3}$$

where α denotes a multi-index and d is the dimension of space. Then, we can obtain the following semi-discrete form by applying weak formulation on each element Ω_e.

Find $\mathbf{U}_h \in V_h^p$ such as

$$\frac{d}{dt}\int_{\Omega e}\mathbf{U_h}\mathbf{W_h}\mathbf{d}\Omega+\int_{\Gamma\mathbf{e}}\mathbf{F_k}(\mathbf{U_h})\mathbf{n_k}\mathbf{W_h}\mathbf{d}\Gamma-\int_{\Omega\mathbf{e}}\mathbf{F_k}(\mathbf{U_h})\frac{\partial\mathbf{W_h}}{\partial\mathbf{x_k}}\mathbf{d}\Omega=\mathbf{0},\quad\forall\mathbf{W_h}\in V_h^p \tag{3.4}$$

where $\mathbf{U}_h$ and $\mathbf{W}_h$ represent the finite element approximations to the analytical solution $\mathbf{U}$ and the test function $\mathbf{W}$ respectively, and they are approximated by piecewise-polynomial functions of degrees p, which are discontinuous between the cell interfaces. Assume that B is the basis of polynomial function of degrees p, this is then equivalent to the following system of N equations,

$$\frac{d}{dt}\int_{\Omega e}\mathbf{U}_hB_i d\Omega+\int_{\Gamma e}\mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_kB_i d\Gamma-\int_{\Omega e}\mathbf{F}_k(\mathbf{U}_h)\frac{\partial B_i}{\partial x_k}d\Omega=0,\quad 1\leqslant i\leqslant \mathrm{N} \tag{3.5}$$

where N is the dimension of the polynomial space. Since the numerical solution $\mathbf{U}_h$ is discontinuous between element interfaces, the interface fluxes are not uniquely defined. The flux function $\mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k$ appearing in the second terms of Eq. (3.5) is replaced by a numerical Riemann flux function $\mathbf{H}_k(\mathbf{U}^L{}_h,\mathbf{U}^R{}_h,\mathbf{n}_k)$ where $\mathbf{U}_h{}^L$ and $\mathbf{U}_h{}^R$ are the conservative state vector at the left and right side of the element boundary. This scheme is called discontinuous Galerkin method of degree p, or in short notation DG(P) method. Note that discontinuous Galerkin formulations are very similar to finite volume schemes, especially in their use of numerical fluxes. Indeed, the classical first-order cell-centered finite volume scheme exactly corresponds to the DG(P_0) method, i.e., to the discontinuous Galerkin method using a piecewise-constant polynomial. Consequently, the DG(P_k) methods with $k > 0$ can be regarded as a natural generalization of finite volume methods to higher order methods. By simply increasing the degree P of the polynomials, the DG methods of corresponding higher order are obtained. The domain and boundary integrals in Eq. (3.5) are calculated using Gauss quadrature formulas. The number of quadrature points used is chosen to integrate exactly polynomials of order of $2p$ and $2p + 1$ for volume and surface inner products in the reference element. In the traditional DGM, termed nodal discontinuous Galerkin methods, numerical polynomial solutions $\mathbf{U}_h$ in each element are expressed using either standard Lagrange finite element or hierarchical node-based basis as following

$$\mathbf{U}_h=\sum_{i=1}^{\mathrm{N}}\mathbf{U}_i(t)B_i(x), \tag{3.6}$$

where B_i are the finite element basis functions. As a result, the unknowns to be solved for are the variables at the nodes $\mathbf{U}_i$.

In the present work, the numerical polynomial solutions are represented using a Taylor series expansion at the center of the cell. For example, if we do a Taylor series expansion at the center of the cell, a quadratic polynomial solution can be expressed as follows

$$\begin{aligned}\mathbf{U}_h=\mathbf{U}_c&+\frac{\partial\mathbf{U}}{\partial x}|_c(x-x_c)+\frac{\partial\mathbf{U}}{\partial y}|_c(y-y_c)+\frac{\partial\mathbf{U}}{\partial z}|_c(z-z_c)+\frac{\partial^2\mathbf{U}}{\partial x^2}|_c\frac{(x-x_c)^2}{2}+\frac{\partial^2\mathbf{U}}{\partial y^2}|_c\frac{(y-y_c)^2}{2}+\frac{\partial^2\mathbf{U}}{\partial z^2}|_c\frac{(z-z_c)^2}{2}\\&+\frac{\partial^2\mathbf{U}}{\partial x\partial y}|_c(x-x_c)(y-y_c)+\frac{\partial^2\mathbf{U}}{\partial x\partial z}|_c(x-x_c)(z-z_c)+\frac{\partial^2\mathbf{U}}{\partial y\partial z}|_c(y-y_c)(z-z_c)\end{aligned} \tag{3.7}$$

which can be further expressed as cell-averaged values and their derivatives at the center of the cell:

$$\begin{aligned}\mathbf{U}_h=\tilde{\mathbf{U}}&+\frac{\partial\mathbf{U}}{\partial x}\bigg|_c(x-x_c)+\frac{\partial\mathbf{U}}{\partial y}\bigg|_c(y-y_c)+\frac{\partial\mathbf{U}}{\partial z}\bigg|_c(z-z_c)+\frac{\partial^2\mathbf{U}}{\partial x^2}\bigg|_c\left(\frac{(x-x_c)^2}{2}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{(x-x_c)^2}{2}d\Omega\right)\\&+\frac{\partial^2\mathbf{U}}{\partial y^2}\bigg|_c\left(\frac{(y-y_c)^2}{2}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{(y-y_c)^2}{2}d\Omega\right)+\frac{\partial^2\mathbf{U}}{\partial z^2}\bigg|_c\left(\frac{(z-z_c)^2}{2}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{(z-z_c)^2}{2}d\Omega\right)\\&+\frac{\partial^2\mathbf{U}}{\partial x\partial y}\bigg|_c\left((x-x_c)(y-y_c)-\frac{1}{\Omega_e}\int_{\Omega_e}(x-x_c)(y-y_c)d\right)+\frac{\partial^2\mathbf{U}}{\partial x\partial z}\bigg|_c\left((x-x_c)(z-z_c)-\frac{1}{\Omega_e}\int_{\Omega_e}(x-x_c)(z-z_c)d\Omega\right)\\&+\frac{\partial^2\mathbf{U}}{\partial y\partial z}\bigg|_c\left((y-y_c)(z-z_c)-\frac{1}{\Omega_e}\int_{\Omega_e}(y-y_c)(z-z_c)d\Omega\right)\end{aligned} \tag{3.8}$$

where $\tilde{\mathbf{U}}$ is the mean value of $\mathbf{U}$ in this cell. The unknowns to be solved for in this formulation are the cell-averaged variables and their derivatives at the center of the cells. The dimension of the polynomial space is 10 and the basis functions are

$$\begin{aligned}&B_1=1,\quad B_2=x-x_c,\quad B_3=y-y_c,\quad B_4=z-z_c\\&B_5=\tfrac{B_2^2}{2}-\tfrac{1}{\Omega_e}\int_{\Omega_e}\frac{B_2^2}{2}d\Omega,\quad B_6=\frac{B_3^2}{2}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_3^2}{2}d\Omega,\quad B_7=\frac{B_4^2}{2}-\frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_4^2}{2}d\Omega,\\&B_8=B_2B_3-\tfrac{1}{\Omega_e}\int_{\Omega_e}B_2B_3d\Omega,\quad B_9=B_2B_4-\frac{1}{\Omega_e}\int_{\Omega_e}B_2B_4d\Omega,\quad B_{10}=B_3B_4-\frac{1}{\Omega_e}\int_{\Omega_e}B_3B_4d\Omega\end{aligned} \tag{3.9}$$

The discontinuous Galerkin formulation then leads to the following 10 equations

$$\begin{aligned}&\frac{d}{dt}\int_{\Omega_e}\tilde{\mathbf{U}}d\Omega+\int_{\Gamma_e}\mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k d\Gamma=0,\quad i=1\\&M_{9\times 9}\frac{d}{dt}\left(\frac{\partial\mathbf{U}}{\partial x}|_c\quad\frac{\partial\mathbf{U}}{\partial y}|_c\quad\frac{\partial\mathbf{U}}{\partial z}|_c\quad\frac{\partial^2\mathbf{U}}{\partial x^2}|_c\quad\frac{\partial^2\mathbf{U}}{\partial y^2}|_c\quad\frac{\partial^2\mathbf{U}}{\partial z^2}|_c\quad\frac{\partial^2\mathbf{U}}{\partial x\partial y}|_c\quad\frac{\partial^2\mathbf{U}}{\partial x\partial z}|_c\quad\frac{\partial^2\mathbf{U}}{\partial y\partial z}|_c\right)^T+R_{9\times 1}=0\end{aligned}\tag{3.10}$$

Note that in this formulation, equations for the cell-averaged variables are decoupled from equations for their derivatives due to the judicial choice of the basis functions and the fact that

$$\int_{\Omega_e}B_1B_i d\Omega=0\qquad 2\leqslant i\leqslant 10\tag{3.11}$$

In the implementation of this DG method, the basis functions are actually normalized in order to improve the conditioning of the system matrix (3.5) as follows:

$$\begin{aligned}&B_1=1,\quad B_2=\tfrac{x-x_c}{\Delta x},\quad B_3=\tfrac{y-y_c}{\Delta y},\quad B_4=\tfrac{z-z_c}{\Delta z}\\&B_5=\tfrac{B_2^2}{2}-\tfrac{1}{\Omega_e}\int_{\Omega_e}\tfrac{B_2^2}{2}d\Omega,\quad B_6=\tfrac{B_3^2}{2}-\tfrac{1}{\Omega_e}\int_{\Omega_e}\tfrac{B_3^2}{2}d\Omega,\quad B_7=\tfrac{B_4^2}{2}-\tfrac{1}{\Omega_e}\int_{\Omega_e}\tfrac{B_4^2}{2}d\Omega,\\&B_8=B_2B_3-\tfrac{1}{\Omega_e}\int_{\Omega_e}B_2B_3d\Omega,\quad B_9=B_2B_4-\tfrac{1}{\Omega_e}\int_{\Omega_e}B_2B_4d\Omega,\quad B_{10}=B_3B_4-\tfrac{1}{\Omega_e}\int_{\Omega_e}B_3B_4d\Omega\end{aligned}\tag{3.12}$$

where $\Delta x = 0.5(x_{\max}-x_{\min})$, $\Delta y = 0.5(y_{\max}-y_{\min})$, and $\Delta z = 0.5(z_{\max}-z_{\min})$, and $x_{\max}$, $x_{\min}$, $y_{\max}$, $y_{\min}$, $z_{\max}$, and $z_{\min}$ are the maximum and minimum coordinates in the cell Ω_e in x-, y-, and z-directions, respectively. A quadratic polynomial solution can then be rewritten as

$$\mathbf{U}_h=\tilde{\mathbf{U}}B_1+\mathbf{U}_xB_2+\mathbf{U}_yB_3+\mathbf{U}_zB_4+\mathbf{U}_{xx}B_5+\mathbf{U}_{yy}B_6+\mathbf{U}_{zz}B_7+\mathbf{U}_{xy}B_8+\mathbf{U}_{xz}B_9+\mathbf{U}_{yz}B_{10}\tag{3.13}$$

where

$$\begin{aligned}&\mathbf{U}_{\mathbf{x}}=\tfrac{\partial\mathbf{U}}{\partial x}|_c\Delta x,\quad\mathbf{U}_y=\tfrac{\partial\mathbf{U}}{\partial y}|_c\Delta y,\quad\mathbf{U}_z=\tfrac{\partial\mathbf{U}}{\partial z}|_c\Delta z,\quad\mathbf{U}_{xx}=\tfrac{\partial^2\mathbf{U}}{\partial x^2}|_c\Delta x^2,\quad\mathbf{U}_{yy}=\tfrac{\partial^2\mathbf{U}}{\partial y^2}|_c\Delta y^2,\\&\mathbf{U}_{zz}=\tfrac{\partial^2\mathbf{U}}{\partial z^2}|_c\Delta z^2,\quad\mathbf{U}_{xy}=\tfrac{\partial^2\mathbf{U}}{\partial x\partial y}|_c\Delta x\Delta y,\quad\mathbf{U}_{xz}=\tfrac{\partial^2\mathbf{U}}{\partial x\partial z}|_c\Delta x\Delta z,\quad\mathbf{U}_{yz}=\tfrac{\partial^2\mathbf{U}}{\partial y\partial z}|_c\Delta y\Delta z\end{aligned}\tag{3.14}$$

This formulation belongs to the so-called modal discontinuous Galerkin method and has a number of attractive, distinct, and useful features. First, cell-averaged variables and their derivatives are handily available in this formulation. This makes the implementation of both in-cell and inter-cell reconstruction schemes straightforward and simple [26–28,30–31]. Secondly, the Taylor basis is hierarchic. This greatly facilitates implementation of p-multigrid methods [16,17] and p-refinement. Thirdly, the same basis functions are used for any shapes of elements: tetrahedron, pyramid, prism, and hexahedron. This makes the implementation of DGM on arbitrary meshes straightforward.

3.2. Least-squares reconstruction

In comparison with reconstructed FV methods, the DGM have a significant drawback in that they require more degrees of freedom, additional domain integration, and more Gauss quadrature points for the boundary integration, and therefore more computational costs and storage requirements. On the one hand, the reconstruction methods that FV methods use to achieve higher-order accuracy are relatively inexpensive but less accurate and robust. On the other hand, the DGM that can be viewed as a different way to extend a FV method to higher orders are accurate and robust but costly. It is only natural and tempting to combine the efficiency of the reconstruction methods and the accuracy of the DG methods. This idea was originally introduced by Dumbser et al in the frame of P_nP_m scheme [18–20], termed RDG(P_nP_m) in this paper, where Pn indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and P_m represents a reconstructed polynomial solution of degree of m ($m \geqslant n$) that is used to compute the fluxes and source terms. The beauty of RDG(P_nP_m) schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of RDG(P_nP_m) schemes, and thus allow for a direct efficiency comparison. When $n = 0$, i.e. a piecewise constant polynomial is used to represent a numerical solution, RDG(P_0P_m) is nothing but classical high order finite volume schemes, where a polynomial solution of degree m ($m \geqslant 1$) is reconstructed from a piecewise constant solution. When $m = n$, the reconstruction reduces to the identity operator, and RDG(P_nP_m) scheme yields a standard DG method. Clearly, an accurate and efficient reconstruction is the key ingredient in extending the underlying DG method to higher order accuracy. Our discussion in this work is mainly focused on the third order RDG(P_1P_2) method, as the benefits of higher-order (>3rd) methods diminish dramatically for engineering applications. Nevertheless, its extension to higher order DG methods is straightforward. In the case of DG(P_1) method, a linear polynomial solution $\mathbf{U}_i$ in any cell i is

$$\mathbf{U}_i=\tilde{\mathbf{U}}_i+\mathbf{U}_{\mathbf{x}i}B_2+\mathbf{U}_{\mathbf{y}i}B_3+\mathbf{U}_{zi}B_4\tag{3.15}$$

Using this underlying linear polynomial DG solution in the neighboring cells, one can reconstruct a quadratic polynomial solution $\mathbf{U}_i^R$ as follows:

$$\mathbf{U}_i^R=\tilde{\mathbf{U}}_i^R+\mathbf{U}_{xi}^RB_2+\mathbf{U}_{yi}^RB_3+\mathbf{U}_{zi}^RB_4+\mathbf{U}_{xxi}^RB_5+\mathbf{U}_{yyi}^RB_6+\mathbf{U}_{zzi}^RB_7+\mathbf{U}_{xyi}^RB_8+\mathbf{U}_{xzi}^RB_9+\mathbf{U}_{yzi}^RB_{10}\tag{3.16}$$

In order to maintain the compactness of the DG methods, the reconstruction is required to involve only von Neumann neighborhood, i.e., the adjacent cells that share a face with the cell i under consideration. There are 10 degrees of freedom, and therefore 10 unknowns must be determined. The first four unknowns can be trivially obtained, by requiring the consistency of the RDG with the underlying DG: (1) The reconstruction scheme must be conservative, and (2) The values of the reconstructed first derivatives are equal to the ones of the first derivatives of the underlying DG solution at the centroid i. Due to the judicious choice of Taylor basis in our DG formulation, these four degrees of freedom simply coincide with the ones from the underlying DG solution, i.e.,

$$\tilde{\mathbf{U}}_i^R = \tilde{\mathbf{U}}_i, \quad \mathbf{U}_{xi}^R = \mathbf{U}_{xi}, \quad \mathbf{U}_{yi}^R = \mathbf{U}_{yi}, \quad \mathbf{U}_{zi}^R = \mathbf{U}_{zi} \tag{3.17}$$

As a result, only six second derivatives need to be determined. This can be accomplished by requiring that the point-wise values and first derivatives of the reconstructed solution and of the underlying DG solution are equal at the cell centers for all the adjacent face neighboring cells. Consider a neighboring cell j, one requires

$$\begin{aligned}
&\mathbf{U}_j = \tilde{\mathbf{U}}_i + \mathbf{U}_{xi}B_2 + \mathbf{U}_{yi}B_3 + \mathbf{U}_{zi}B_4 + \mathbf{U}_{xxi}^R B_5 + \mathbf{U}_{yyi}^R B_6 + \mathbf{U}_{zzi}^R B_7 + \mathbf{U}_{xyi}^R B_8 + \mathbf{U}_{xzi}^R B_9 + \mathbf{U}_{yzi}^R B_{10} \\
&\frac{\partial \mathbf{U}}{\partial x}\Big|_j = \mathbf{U}_{xi}\frac{1}{\Delta x_i} + \mathbf{U}_{xxi}^R\frac{B_2}{\Delta x_i} + \mathbf{U}_{xyi}^R\frac{B_3}{\Delta x_i} + \mathbf{U}_{xzi}^R\frac{B_4}{\Delta x_i} \\
&\frac{\partial \mathbf{U}}{\partial y}\Big|_j = \mathbf{U}_{yi}\frac{1}{\Delta y_i} + \mathbf{U}_{yyi}^R\frac{B_3}{\Delta y_i} + \mathbf{U}_{xyi}^R\frac{B_2}{\Delta y_i} + \mathbf{U}_{yzi}^R\frac{B_4}{\Delta y_i} \\
&\frac{\partial \mathbf{U}}{\partial z}\Big|_j = \mathbf{U}_{zi}\frac{1}{\Delta z_i} + \mathbf{U}_{zzi}^R\frac{B_4}{\Delta z_i} + \mathbf{U}_{xzi}^R\frac{B_2}{\Delta z_i} + \mathbf{U}_{yzi}^R\frac{B_3}{\Delta z_i}
\end{aligned} \tag{3.18}$$

where the basis functions B are evaluated at the center of cell j, i.e., $B = \mathbf{B}(x_j, y_j, z_j)$. This can be written in a matrix form as follows:

$$\begin{pmatrix} B_5^j & B_6^j & B_7^j & B_8^j & B_9^j & B_{10}^j \\ B_2^j & 0 & 0 & B_3^j & B_4^j & 0 \\ 0 & B_3^j & 0 & B_2^j & 0 & B_4^j \\ 0 & 0 & B_4^j & 0 & B_2^j & B_3^j \end{pmatrix} \begin{pmatrix} \mathbf{U}_{xxi}^R \\ \mathbf{U}_{yyi}^R \\ \mathbf{U}_{zzi}^R \\ \mathbf{U}_{xyi}^R \\ \mathbf{U}_{xzi}^R \\ \mathbf{U}_{yzi}^R \end{pmatrix} = \begin{pmatrix} \mathbf{U}_j - (\mathbf{U}_i B_1^j + \mathbf{U}_{xi}B_2^j + \mathbf{U}_{yi}B_3^j + \mathbf{U}_{zi}B_4^j) \\ \frac{\Delta x_i}{\Delta x_j}\mathbf{U}_{xj} - \mathbf{U}_{xi} \\ \frac{\Delta y_i}{\Delta y_j}\mathbf{U}_{yj} - \mathbf{U}_{yi} \\ \frac{\Delta z_i}{\Delta z_j}\mathbf{U}_{zj} - \mathbf{U}_{zi} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1^j \\ \mathbf{R}_2^j \\ \mathbf{R}_3^j \\ \mathbf{R}_4^j \end{pmatrix} \tag{3.19}$$

where $\mathbf{R}$ is used to represent the right-hand-side for simplicity. Similar equations can be written for all cells connected to the cell i with a common face, which leads to a non-square matrix. The number of face-neighboring cells for a tetrahedral is four. Consequently, the size of the resulting non-square matrix is 16×6. In the present work, this over-determined linear system of 16 equations for 6 unknowns is solved in the least-squares sense using both normal equation approach and the QR decomposition to obtain the second derivatives of the reconstructed quadratic polynomial solution. One can easily verify that this least-squares reconstruction satisfies the so-called 2-exactness, i.e., it can reconstruct a quadratic polynomial function exactly.

3.3. Hermite WENO reconstruction

This least-squares reconstructed discontinuous Galerkin method: RDG(P_1P_2) has been successfully used to solve the 2D compressible Euler equations on arbitrary grids [26–28,30–31] and is able to achieve the designed third order of accuracy and significantly improve the accuracy of the underlying second-order DG method. However, when used to solve the 3D compressible Euler equations on tetrahedral grids, this RDG method suffers from the so-called linear instability, that is also observed in the second-order cell-centered finite volume methods, i.e., RDG(P_0P_1) [33]. This linear instability is attributed to the fact that the reconstruction stencils only involve von Neumann neighborhood, i.e., adjacent face-neighboring cells [33]. The linear stability can be achieved using extended stencils, which will unfortunately sacrifice the compactness of the underlying DG methods. Furthermore, such a linear reconstruction-based DG method suffers from non-linear instability, leading to non-physical oscillations in the vicinity of strong discontinuities. Alternatively, ENO, WENO, and HWENO can be used to reconstruct a higher-order polynomial solution, which can not only enhance the order of accuracy of the underlying DG method but also achieve both linear and non-linear stability. In the present work, the reconstructed quadratic polynomial based on the Hermite WENO on cell i are a convex combination of the least-squares reconstructed second derivatives at the cell itself and its four face-neighboring cells,

$$\left.\frac{\partial^2 U}{\partial x_m \partial x_n}\right|_i = \sum_{k=1}^{5} w_k \left.\frac{\partial^2 U}{\partial x_m \partial x_n}\right|_k \tag{3.20}$$

where, the weights w_k are computed as

$$w_k = \frac{(\varepsilon + o_k)^{-\gamma}}{\sum_{i=1}^{5}(\varepsilon + o_i)^{-\gamma}} \tag{3.21}$$

Table 1
Cost analysis for different numerical methods on a tetrahedral grid.

	RDG(P_0P_1)	RDG(P_1P_1)	RDG(P_1P_2)	RDG(P_2P_2)
Number of quadrature points for boundary integrals	1	3	4	7
Number of quadrature points for domain integrals	0	4	5	11
Reconstruction	Yes	No	Yes	No
Order of accuracy	$O(h^2)$	$O(h^2)$	$O(h^3)$	$O(h^3)$
Storage for implicit diagonal matrix	25 words per element	400	400	2500

where ε is a small positive number used to avoid division by zero, o_k the oscillation indicator for the reconstructed second order polynomials, and γ an integer parameter to control how fast the non-linear weights decay for non-smooth stencils. The oscillation indicator is defined as

$$o_k = \left[\int_{\Omega_i}\left(\frac{\partial^2 U}{\partial x_m \partial x_n}|_k\right)^2 d\Omega\right]^2 \tag{3.22}$$

Note that the least-squares reconstructed polynomial at the cell itself serves as the central stencil and the least-squares reconstructed polynomials on its four face-neighboring cells act as biased stencils in this WENO reconstruction. This reconstructed quadratic polynomial solution is then used to compute the domain and boundary integrals of the underlying DG(P_1) method in Eq. (3.5). The resulting DG method, termed a "reconstructed DG" method (RDG(P_1P_2) in short notation), is expected to have third order of accuracy at a moderate increase of computing costs in comparison to the underlying DG(P_1) method. The extra costs are mainly due to the least-squares reconstruction, which is relatively cheap in comparison to the evaluation of fluxes, and an extra Gauss quadrature point, which is required to calculate both domain and boundary integrals. In comparison to DG(P_2), this represents a significant saving in terms of flux evaluations. Furthermore, the number of degrees of freedom is considerably reduced, which leads to a significant reduction in memory requirements, and from which implicit methods will benefit tremendously. The cost analysis for the RDG(P_0P_1) (FV(P_1)), RDG(P_1P_1) (DG(P_1)), RDG(P_1P_2) and RDG(P_2P_2) (DG(P_2)) is summarized in Table 1, where the memory requirement for storing only the implicit diagonal matrix is given as well, and which grows quadratically with the order of the DG methods. We would like to emphasize that the storage requirements for the implicit DG methods are extremely demanding, especially for higher-order DG methods.

As demonstrated in the next section, this RDG method is able to achieve the required linear stability and the designed third order of accuracy. Even though it does not introduce any new oscillatory behavior due to the WENO reconstruction, it cannot remove inherent oscillations in the underlying DG(P_1) solutions. Consequently, it still suffers from the non-linear instability for flows with strong discontinuities. However, the non-linear instability can be obtained by conducting a WENO reconstruction on the first order derivatives using a hierarchical reconstruction [35,15], which will be reported in a follow-up manuscript.

4. Numerical examples

The Hermite WENO reconstruction method has been implemented in a well-tested DG code [13–17] to solve a variety of the compressible flow problems on tetrahedral grids. A fast *p*-multigrid method [16,17] is used to obtain steady state solutions. A few examples are presented in this section to demonstrate that the developed RDG(P_1P_2) method is able to maintain the linear stability, achieve the designed 3rd order of accuracy, and significantly improve the accuracy of the underlying second-order DG method without significant increase in computing costs and storage requirements.

4.1. A convergence study for the quadratic Hermite WENO reconstruction

The first test case is chosen to validate the implementation of the developed Hermite WENO reconstruction scheme and to demonstrate numerically its 3rd order of convergence on tetrahedral grids. In order to obtain a quantitative measurement of the order of accuracy and reconstruction errors, the designed Hermite WENO reconstruction is conducted for a smooth function $f(x,y,z) = \sin(\pi x)\cos(2\pi y)\sin(3\pi z)$ on a cubic domain $[0,1]\times[0,1]\times[0,1]$ on four successfully refined grids as shown in Fig. 1. The numbers of elements, points, and boundary points for these four grids are (547,156,103), (4406,990,444), (35697,6973,1785), and (286702,52093,7094), respectively. The L_2-norm of the error between the analytical and reconstructed functions is used as the error measurement. The errors and the convergence rates are reported in Table 2. It shows the mesh size, the L_2-error of the error function, and the order of convergence, where one can see that the developed Hermite WENO reconstruction indeed is able to deliver the designed 3rd order of convergence.

4.2. A subsonic flow through a channel with a smooth bump

This test case is chosen to demonstrate the convergence rates of the RDG(P_0P_1), RDG(P_1P_1), and RDG(P_1P_2) methods for internal flows. The problem under consideration is a subsonic flow inside a 3D channel with a smooth bump on the lower

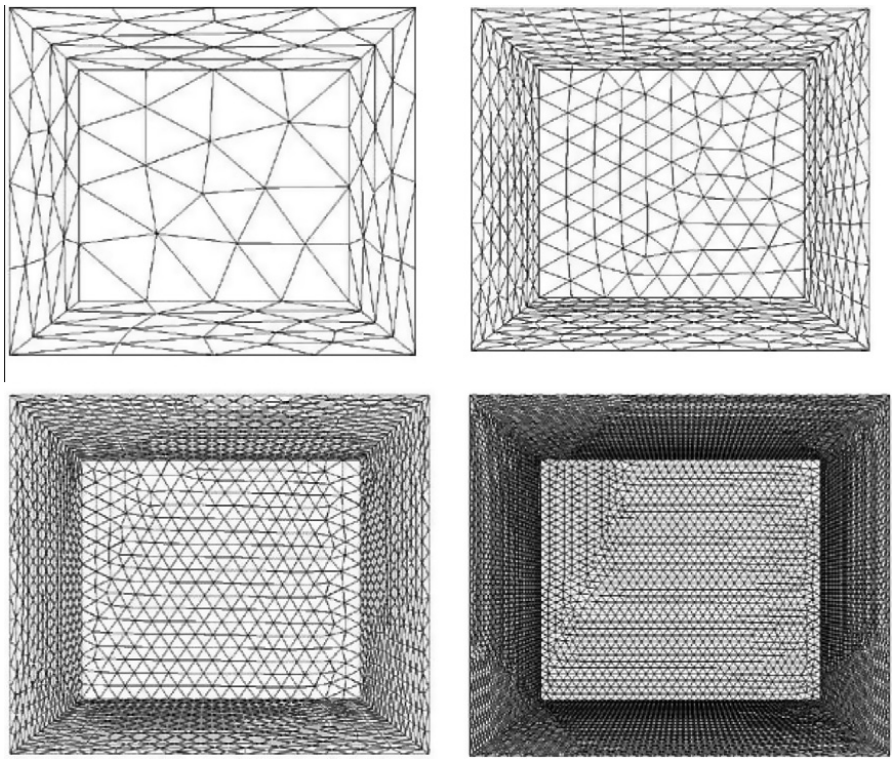

Fig. 1. Sequences of the four successively globally refined tetrahedral meshes used to study the convergence of the Hermite WENO reconstruction.

Table 2
L_2-error and order of the convergence for the quadratic Hermite WENO reconstruction.

Number of cells	L_2-error	Order
547	3.44033E-2	-
4,406	4.24326E-03	3.01
35,697	4.46581E-04	3.23
286,705	4.93515E-05	3.17

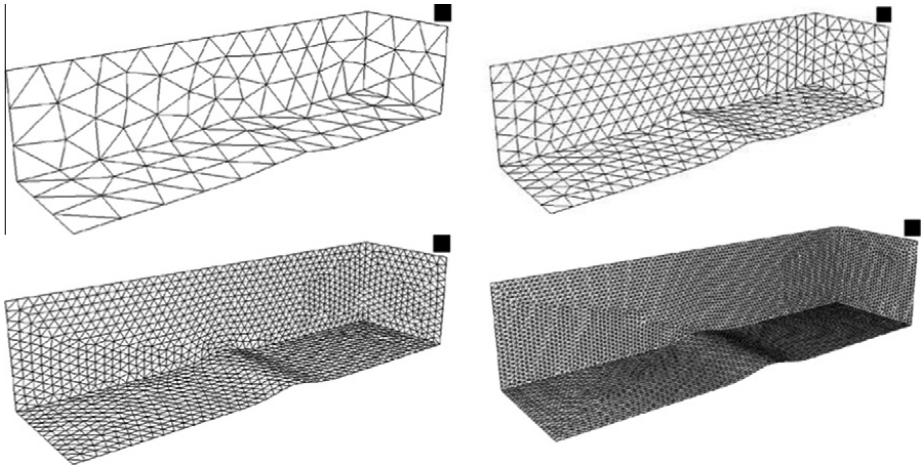

Fig. 2. A sequence of four successively globally refined unstructured meshes used for computing subsonic flow in a channel with a smooth bump.

surface. The height, width, and length of the channel are 0.8, 0.8, and 3, respectively. The shape of the lower wall is defined by the function $0.0625\exp(-25x^2)$ from $x = -1.5$ to $x = 1.5$. The inflow condition is prescribed at a Mach number of 0.5, and an angle of attack of 0°. Fig. 2 shows the four successively refined tetrahedral grids used for the grid convergence study. The

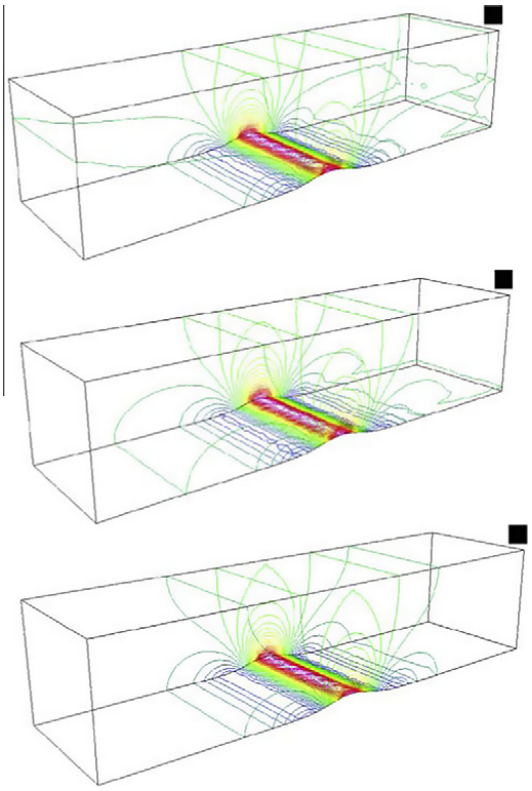

Fig. 3. Computed velocity contours in the flow field obtained by the RDG(P_0P_1) on the finest grid (top) and RDG(P_1P_1) (middle), and RDG(P_1P_2) (bottom) for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$.

numbers of elements, points, and boundary points for the coarse, medium, fine, and finest grids are (889,254,171), (6986,1555,691), (55703,10822,2711), and (449522,81567,10999), respectively. The cell size is halved between consecutive meshes. Numerical solutions to this problem are computed using the RDG(P_1P_1) and RDG(P_1P_2) methods on the first three grids and the RDG(P_0P_1) on the last three grids to obtain a quantitative measurement of the order of accuracy and discretization errors. The following L_2-norm of the entropy production is used as the error measurement

$$\|\varepsilon\|_{L_2(\Omega)} = \sqrt{\int_\Omega \varepsilon^2 d\Omega}$$

where the entropy production ε defined as

$$\varepsilon = \frac{S - S_\infty}{S_\infty} = \frac{p}{p_\infty}\left(\frac{\rho_\infty}{\rho}\right)^\gamma - 1$$

Note that the entropy production, where the entropy is defined as $S = p/\rho^\gamma$, is a very good criterion to measure accuracy of the numerical solutions, since the flow under consideration is isentropic. Fig. 3 illustrates the computed velocity contours in the flow field obtained by the RDG(P_0P_1) method on the finest grid (449522,81567,10999), and the RDG(P_1P_1) and RDG(P_1P_2) methods on the fine grid (55703,10822,2711). One can observe that the second-order DG method (RDG(P_1P_1)) on the fine grid provides a more accurate solution than the second-order finite volume method (RDG(P_0P_1)) on the finest grid, and the RDG(P_1P_2) solution is much more accurate than the RDG(P_1P_1) solution on the same fine grid. Fig. 4 provides the details of the spatial convergence of the three RDG methods for this numerical experiment. Consider the fact that this is a 3D simulation of a 2D problem, and unstructured tetrahedral grids are not symmetric by nature, thus causing error in the z-direction, the second order RDG(P_1P_1) method can be considered to deliver the designed second order of accuracy. As expected, the DG(P_1P_2) method offers a full O(h^{p+2}) order of the convergence, adding one order of accuracy to the underlying DG(P_1) method. Although the RDG(P_0P_1) is designed to be second order accurate, it only yields an order of 1.47 for this test case. In reality, the second order finite volume method RDG(P_0P_1) can hardly reach the designed second order of convergence for solving the compressible Euler equations on tetrahedral grids. Note that in order to reach the same level of the error (-3.6), the number of degrees of freedom required by the second-order RDG(P_0P_1) is one order of magnitude more than the one required by the third order RDG(P_1P_2) method, demonstrating that the RDG(P_1P_2) method can significantly increase the accuracy of the underlying DG method, and therefore greatly decrease its computing costs.

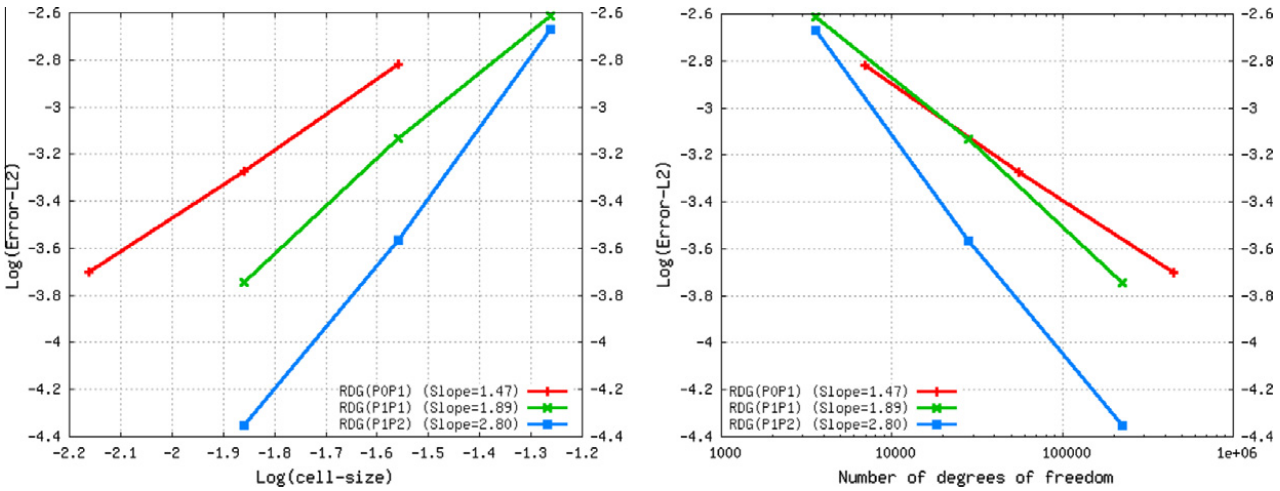

Fig. 4. Convergence history for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$ for different reconstructed RDG methods.

4.3. Subsonic flows past a sphere

In this test case, a subsonic flow past a sphere at a Mach number of $M_\infty = 0.5$ is chosen to assess if the developed RDG method can achieve a formal order of the convergence rate for external flows. A sequence of the four successively refined tetrahedral grids used in this grid convergence study is shown in Fig. 5. The numbers of elements, points, and boundary points for the coarse, medium, fine, and finest grids are (535,167,124), (2426,598,322), (16467,3425,1188), and (124706,23462,4538), respectively. The cell size is halved between consecutive meshes. Note that only a quarter of the configuration is modeled due to the symmetry of the problem, and that the number of elements on a successively refined mesh is not exactly eight times the coarse mesh's elements due to the nature of unstructured grid generation. The computations are conducted on the first three grids using the RDG(P_1P_1) and RDG(P_1P_2) methods and the last three grids using the RDG(P_0P_1) method. As in the previous case, the entropy production is used as the error measurement. Fig. 6 illustrates the computed velocity contours in the flow field obtained by the RDG(P_0P_1) method on the finest grid and the RDG(P_1P_1)

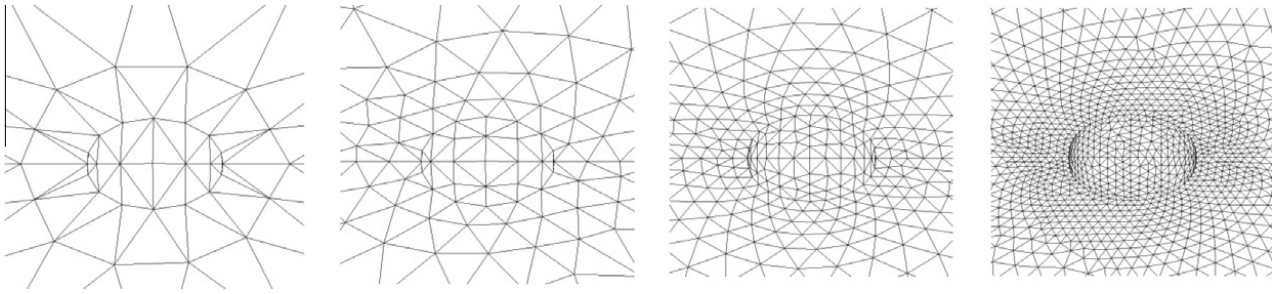

Fig. 5. A series of four successively globally refined tetrahedral meshes for computing subsonic flow past a sphere at $M_\infty = 0.5$.

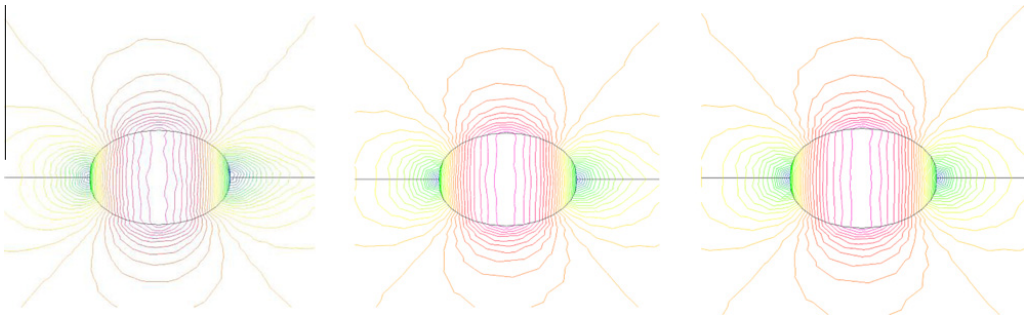

Fig. 6. Computed velocity contours in the flow field obtained by the RDG(P_0P_1) method on the finest mesh (left), RDG(P_1P_1) on the fine mesh (middle), and RDG(P_1P_2) on the fine mesh (right) for subsonic flow past a sphere at $M_\infty = 0.5$.

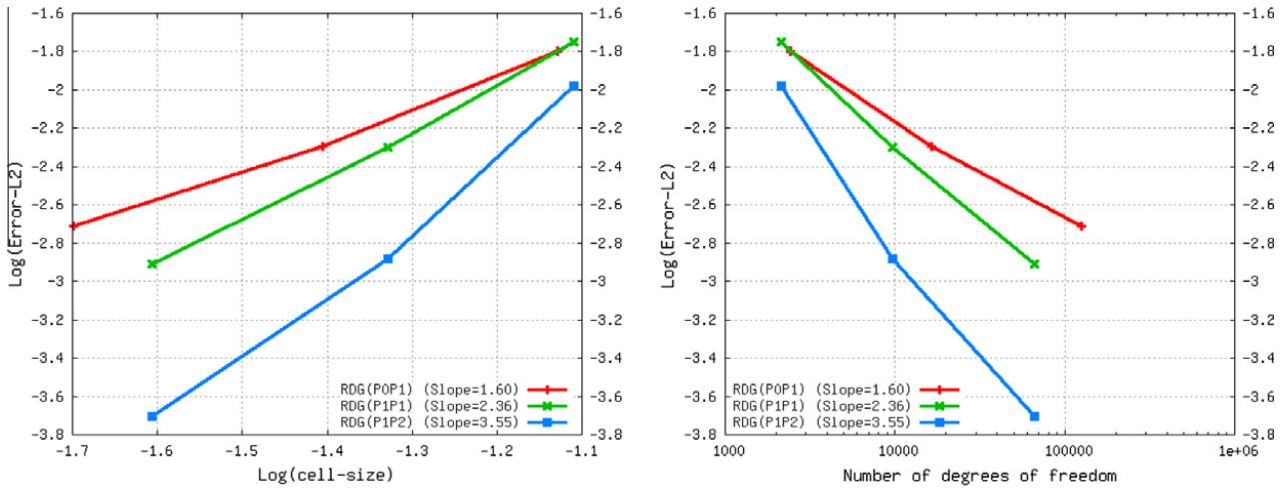

Fig. 7. Convergence history for a subsonic flow past a sphere at $M_\infty = 0.5$ for different reconstructed RDG methods.

and RDG(P_1P_2) methods on the fine grid. One can observe that the second-order DG method (RDG(P_1P_1)) on the fine grid yields a more accurate solution than the second-order finite volume method (RDG(P_0P_1)) on the finest grid, and the RDG(P_1P_2) solution is much more accurate than the RDG(P_1P_1) solution on the same fine grid. Fig. 7 provides the details of the spatial convergence of the three RDG methods for this numerical experiment. Both RDG(P_1P_1) and RDG(P_1P_2) methods achieve higher than expected convergence rates, being 2.36 and 3.55 respectively, and the RDG(P_0P_1) fails to deliver the designed second order accuracy, achieving only an order of 1.6 for this test case. To achieve an error level below -3 in Fig. 7, the RDG(P_0P_1) method needs one order of magnitude more degrees of freedom than the RDG(P_1P_2) method, convincingly demonstrating the benefits of using a higher-order method.

In this test case, a subsonic flow past a sphere at a Mach number of $M_\infty = 0.5$ is chosen to assess if the developed RDG method can achieve a formal order of the convergence rate for external flows. A sequence of the four successively refined tetrahedral grids used in this grid convergence study is shown in Fig. 5. The numbers of elements, points, and boundary points for the coarse, medium, fine, and finest grids are (535,167,124), (2426,598,322), (16467,3425,1188), and (124706,23462,4538), respectively. The cell size is halved between consecutive meshes. Note that only a quarter of the configuration is modeled due to the symmetry of the problem, and that the number of elements on a successively refined mesh is not exactly eight times the coarse mesh's elements due to the nature of unstructured grid generation. The computations are conducted on the first three grids using the RDG(P_1P_1) and RDG(P_1P_2) methods and the last three grids using the RDG(P_0P_1) method. As in the previous case, the entropy production is used as the error measurement. Fig. 6 illustrates the computed velocity contours in the flow field obtained by the RDG(P_0P_1) method on the finest grid and the RDG(P_1P_1) and RDG(P_1P_2) methods on the fine grid. One can observe that the second-order DG method (RDG(P_1P_1)) on the fine grid

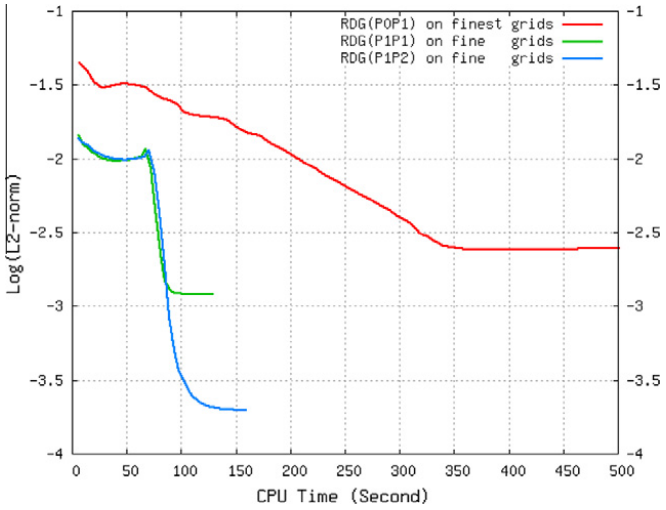

Fig. 8. Numerical error versus CPU time for different reconstructed RDG methods.

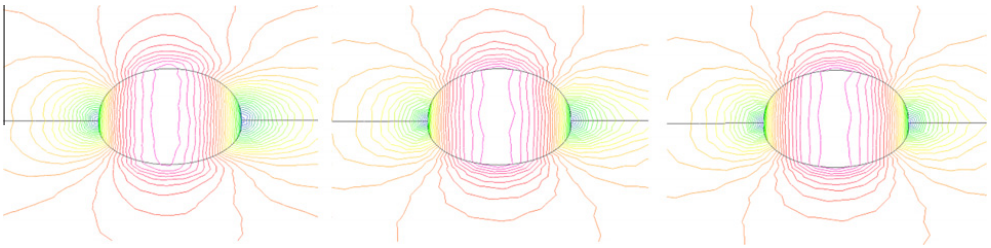

Fig. 9. Computed velocity contours in the flow field obtained by the RDG(P_0P_1) method on the finest mesh (left), RDG(P_1P_1) on the fine mesh (middle), and RDG(P_1P_2) on the fine mesh (right) for a low Mach number flow past a sphere at $M_\infty = 0.5$.

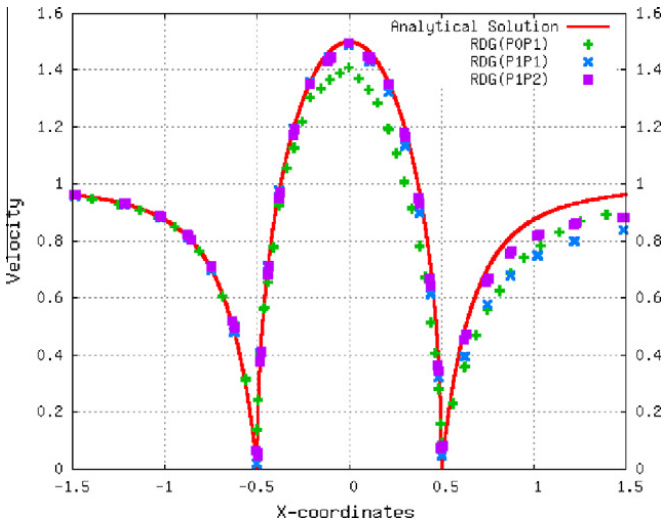

Fig. 10. Comparison of the computed velocity distributions on the surface of the sphere obtained by the RDG(P_0P_1) method on the finest grid and the RDG(P_1P_1) and RDG(P_1P_2) methods on the fine grid with the incompressible solution for a low-Mach-number flow past a sphere at $M_\infty = 0.01$.

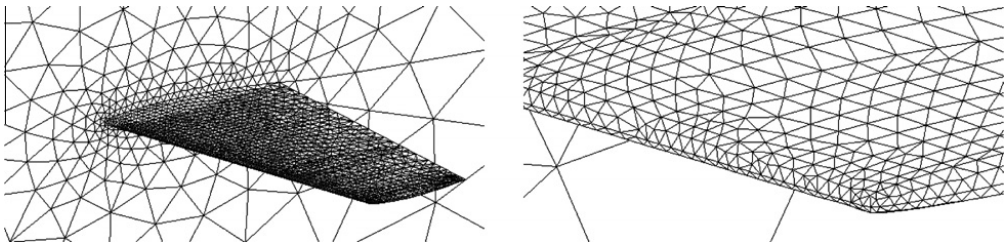

Fig. 11. Unstructured mesh used for computing a transonic flow past an ONERA M6 wing (41,440 elements, 8325 points, 2575 boundary points).

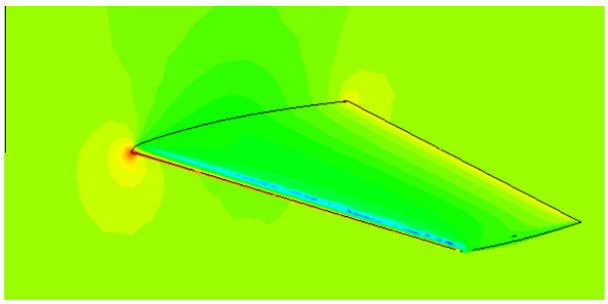

Fig. 12. Computed pressure contours obtained by the RDG(P_1P_2) method for transonic flow past a M6 wing at $M_\infty = 0.699$, $\alpha = 3.06°$.

yields a more accurate solution than the second-order finite volume method (RDG(P_0P_1)) on the finest grid, and the RDG(P_1P_2) solution is much more accurate than the RDG(P_1P_1) solution on the same fine grid. Fig. 7 provides the details

of the spatial convergence of the three RDG methods for this numerical experiment. Both RDG(P_1P_1) and RDG(P_1P_2) methods achieve higher than expected convergence rates, being 2.36 and 3.55 respectively, and the RDG(P_0P_1) fails to deliver the designed second order accuracy, achieving only an order of 1.6 for this test case. Fig. 8 shows the plot of the numerical error against the CPU time required for the RDG(P_1P_2) and RDG(P_1P_1) methods on the fine grid and the RDG(P_0P_1) method on the finest grid. As can be clearly seen from Fig. 8, the RDG(P_1P_2) method provides a significant saving (probably orders of magnitude when a high accuracy is requested.) in CPU time in comparison with the second-order finite volume method (RDG(P_1P_0)), convincingly demonstrating the superior performance of the RDG(P_1P_2) method. Indeed, as indicated in Fig. 7, the RDG(P_0P_1) method needs one order of magnitude more degrees of freedom, and consequently CPU time than the RDG(P_1P_2) method in order to achieve an error level below -3.

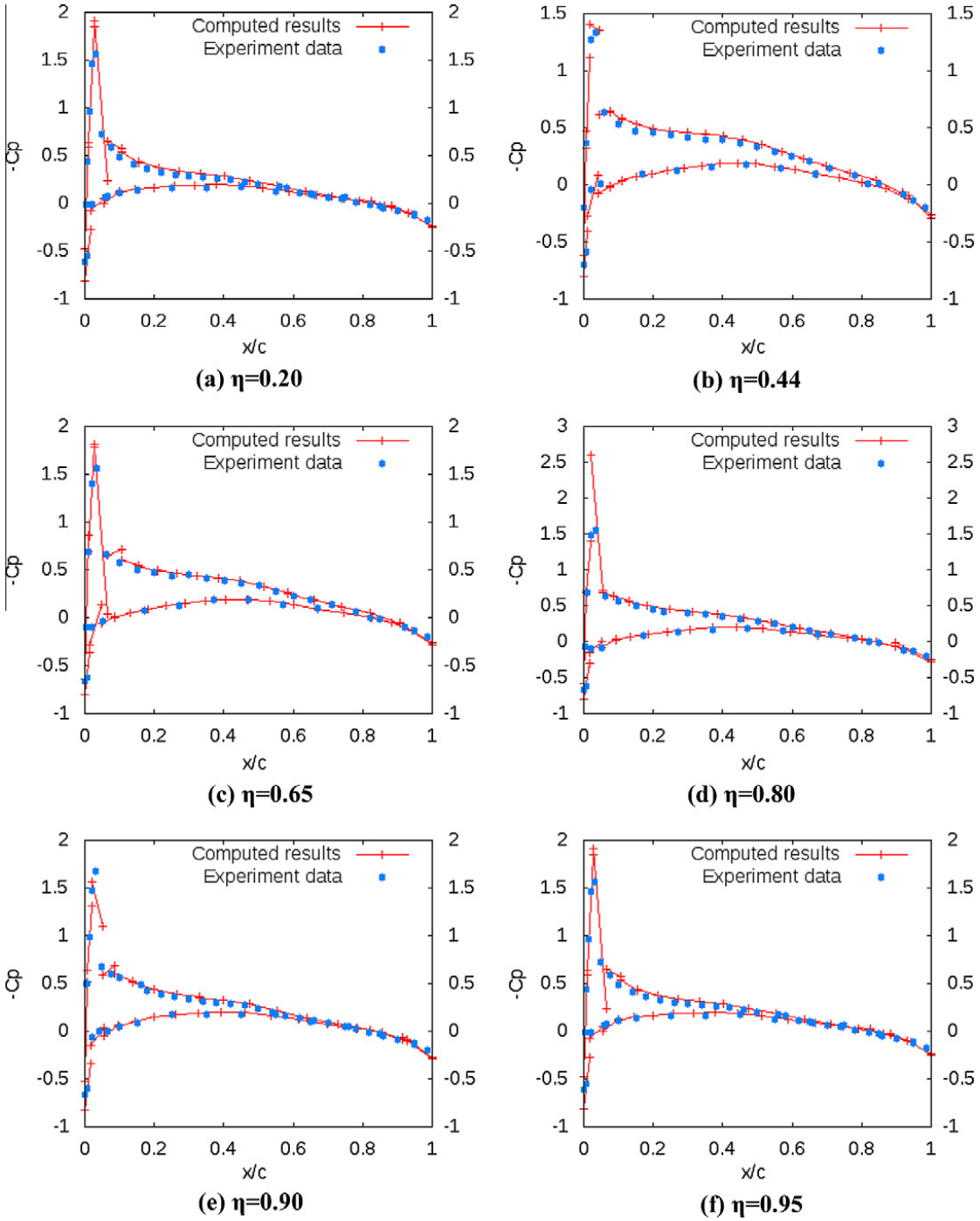

Fig. 13. Comparison of the computed pressure coefficient distributions with experimental data for ONERA M6 wing at 6 span-wise locations, $M_\infty = 0.99$, $\alpha = 3.06°$.

4.4. Low Mach number flow past a sphere

This test case is selected to demonstrate that the RDG method is able to enhance the accuracy of the underlying DG method for solving low Mach number flow problems. The computation is performed for a low Mach number flow past a sphere at a Mach number of $M_\infty = 0.01$ using the RDG(P_0P_1) method on the finest grid and the RDG(P_1P_1), and RDG(P_1P_2) methods on the fine grid in the previous test case. The computed velocity contours in the flow field obtained by these three methods are shown in Fig. 9, respectively. Fig. 10 compares the velocity distributions on the surface of the sphere obtained by these three methods with the one from the incompressible potential flow solution. The deterioration of the RDG(P_0P_1) solution is clearly visible. The RDG(P_1P_2) solution is even more accurate than the RDG(P_1P_1) solution, therefore demonstrating that the RDG(P_1P_2) method is able to maintain the ability of the underlying DG method for accurately computing low Mach number flows.

4.5. Transonic flow past an ONERA M6 wing

A transonic flow over the ONERA M6 wing at a Mach number of $M_\infty = 0.699$ and an attack angle of $\alpha = 3.06^\circ$ is considered in this example. This test case is chosen to demonstrate that the RDG(P_1P_2) method is able to maintain the robustness of the underlying methods. The DG method is not only linear stable but also has the ability to obtain a stable solution for weak discontinuities in spite of the over- and under-shots in the vicinity of shock waves. The mesh used in this computation consists of 41,440 elements, 8325 grid points, and 2575 boundary points, as shown in Fig. 11, where one can observe the coarseness of grids even in the vicinity of the leading edge. The computed pressure contours obtained by the RDG(P_1P_2) solution on the wing surface are shown in Fig. 12. Fig. 13 compares the pressure coefficient distributions at six span-wise locations on the wing surface between the numerical results and the experimental data. The pressure coefficients are computed at the two nodes of each triangle that intersect with the cut plane, and plotted by a straight line. This representation truly reflects the discontinuous nature of the DG solution. Since no limiters and WENO-reconstruction are used to eliminate the spurious oscillations for the underlying DG(P_1) method, the over- and under-shoots in the vicinity of the shock waves are clearly visible. Nevertheless, this example clearly demonstrates that the RDG(P_1P_2) is able to maintain the linear stability of the underlying DG method.

5. Conclusions

A discontinuous Galerkin method based on a Hermite WENO reconstruction has been presented for solving the compressible Euler equations on tetrahedral grids. The nonlinear Hermite WENO reconstruction, necessary to maintain a linear stability of the RDG method, is designed not only to enhance the accuracy of the discontinuous Galerkin method, but also to avoid non-physical oscillations in the vicinity of discontinuities. A number of numerical experiments for a variety of flow conditions have been conducted to demonstrate the superior performance of this RDG(P_1P_2) method over the underlying DG(P_1) method and to assess its convergence rates. The presented numerical results indicate that this Hermite WENO reconstruction-based RDG(P_1P_2) method is able to achieve the designed third-order of accuracy: one order accuracy higher than the underlying DG method, and thus significantly increase the accuracy of the underlying DG method, without significant increase in computing costs and memory requirements. The developed Hermite WENO reconstruction-based RDG method has also successfully been extended to flows with strong discontinuities using a hierarchical reconstruction and will be reported in a follow-up paper.

Acknowledgments

This manuscript has been authored by Battelle Energy Alliance, LLC under Contract No. DE-AC07-05ID14517 (INL/CON-09-16528) with the U.S. Department of Energy. The United States Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The authors would like to acknowledge the partial support for this work provided by DOE under Nuclear Engineering University Program, by the NSF under Project No. NSF-DMS0914706. The first author would like to acknowledge the partial support for this work provided by the fundamental research program of DTRA under Grant No. HDTR1-10-1-0.123. Dr. Suhithi Peiris serves as the technical monitor.

References

[1] W.H. Reed, T.R. Hill, Triangular Mesh Methods for the Neutron Transport Equation, Los Alamos Scientific Laboratory, Report, LA-UR-73-479, 1973.
[2] B. Cockburn, S. Hou, C.W. Shu, TVD Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for conservation laws IV: the Multidimensional Case, Mathematics of Computation, Vol. 55, pp. 545–581, 1990. George, P. L., Automatic Mesh Generation, J. Wiley & Sons, 1991.
[3] B. Cockburn, C.W. Shu, The Runge–Kutta discontinuous Galerkin method for conservation laws V: Multidimensional system, Journal of Computational Physics 141 (1998) 199–224.
[4] B. Cockburn, G. Karniadakis, C.W. Shu, The Development of Discontinuous Galerkin Method, in Discontinuous Galerkin Methods, Theory, Computation, and Applications, in: B. Cockburn, G.E. Karniadakis, C.W. Shu (Eds.), Lecture Notes in Computational Science and Engineering, Springer-Verlag, New York, 2000, Vol. 11, pp. 5–50, 2000.

[5] F. Bassi, S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, Journal of Computational Physics 138 (1997) 251–285.
[6] H.L. Atkins, C.W. Shu, Quadrature Free Implementation of Discontinuous Galerkin Method for Hyperbolic Equations, AIAA Journal, Vol. 36, No. 5, 1998.
[7] F. Bassi, S. Rebay, GMRES discontinuous Galerkin solution of the Compressible Navier–Stokes Equations, Discontinuous Galerkin Methods, Theory, Computation, and Applications, B. Cockburn, G.E. Karniadakis, C.W. Shu (Eds.), Lecture Notes in Computational Science and Engineering, Springer-Verlag, New York, 2000, Vol. 11 pp. 197–208, 2000.
[8] T.C. Warburton, G.E. Karniadakis, A Discontinuous Galerkin method for the viscous MHD equations, Journal of Computational Physics 152 (1999) 608–641.
[9] J.S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, Texts in, Applied Mathematics, Vol. 56, 2008.
[10] P. Rasetarinera, M.Y. Hussaini, An efficient implicit discontinuous spectral Galerkin method, Journal of Computational Physics 172 (2001) 718–738.
[11] B.T. Helenbrook, D. Mavriplis, H.L. Atkins, Analysis of p-Multigrid for Continuous and Discontinuous Finite Element Discretizations, AIAA Paper 2003–3989, 2003.
[12] K.J. Fidkowski, T.A. Oliver, J. Lu, D.L. Darmofal, p-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations, Journal of Computational Physics 207 (1) (2005) 92–113.
[13] H. Luo, J.D. Baum, R. Löhner, A discontinuous Galerkin method using taylor basis for compressible flows on arbitrary grids, Journal of Computational Physics 227 (20) (October 2008) 8875–8893.
[14] H. Luo, J.D. Baum, R. Löhner, "On the Computation of Steady-State Compressible Flows Using a Discontinuous Galerkin Method", International Journal for Numerical Methods in Engineering 73 (5) (2008) 597–623.
[15] H. Luo, J.D. Baum, R. Löhner, A Hermite WENO-based limiter for discontinuous galerkin method on unstructured grids, Journal of Computational Physics 225 (1) (2007) 686–713.
[16] H. Luo, J.D. Baum, R. Löhner, A *p*-multigrid discontinuous Galerkin method for the euler equations on unstructured grids, Journal of Computational Physics 211 (2) (2006) 767–783.
[17] H. Luo, J.D. Baum, R. Löhner, Fast, *p*-multigrid discontinuous Galerkin method for compressible flows at all speeds, AIAA Journal 46 (3) (2008) 635–652.
[18] M. Dumbser, D.S. Balsara, E.F. Toro, C.D. Munz, A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes, Journal of Computational Physics 227 (2008) 8209–8253.
[19] M. Dumbser, O. Zanotti, Very high order P_nP_m schemes on unstructured meshes for the resistive relativistic MHD equations, Journal of Computational Physics 228 (2009) 6991–7006.
[20] M. Dumbser, Arbitrary high order P_nP_m schemes on unstructured meshes for the compressible Navier–Stokes equations, Computers & Fluids 39 (2010) 60–76.
[21] F. Bassi, S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations, Journal of Computational Physics 131 (1997) 267–279.
[22] F. Bassi, S. Rebay, Discontinuous Galerkin solution of the Reynolds–averaged Navier–Stokes and k–ω turbulence model equations, Journal of Computational Physics 34 (2005) 507–540.
[23] B. Cockburn, C.W. Shu, The local discontinuous Galerkin method for time-dependent convection–diffusion system, SIAM, Journal of Numerical Analysis, Vol. 16, 2001.
[24] C.E. Baumann, J.T. Oden, A discontinuous hp finite element method for the Euler and Navier–Stokes equations, International Journal for Numerical Methods in Fluids, Vol. 31, 1999.
[25] H. Luo, L. Luo, k. Xu, A discontinuous Galerkin method based on a BGK scheme for the Navier–Stokes equations on arbitrary grids, Advances in Applied Mathematics and Mechanics 1 (3) (2009) 301–318.
[26] H. Luo, L. Luo, R. Nourgaliev, A Reconstructed Discontinuous Galerkin Method for the Euler Equations on Arbitrary Grids, Communication in Computational Physics 12 (5) (2012) 1495–1519.
[27] H. Luo, L. Luo, R. Norgaliev, V.A. Mousseau, N. Dinh, A reconstructed discontinuous Galerkin method for the compressible Navier–Stokes equations on arbitrary grids, Journal of Computational Physics 229 (2010) 6961–6978.
[28] H. Luo, L. Luo, A. Ali, R. Nourgaliev, C. Cai, A parallel, reconstructed discontinuous Galerkin method for the compressible flows on arbitrary grids, Communication in Computational Physics 9 (2) (2011) 363–389.
[29] L.P. Zhang, W. Liu, L.X. He, X.G. Deng, H.X. Zhang, A Class of Hybrid DG/FV Methods for Conservation Laws I: Basic Formulation and One-Dimensional System, doi:10.1016/j.jcp. 2011.06.010, Journal of Computational Physics.
[30] Zhang L.P., Liu W., He L.X., Deng, X.G., Zhang H.X., A Class of Hybrid DG/FV Methods for Conservation Laws II: Two dimensional Cases, doi:10.1016/j.jcp. 2011.03.032, Journal of Computational Physics, 2011.
[31] H. Luo, Y. Xia, R. Nourgaliev, C. Cai, A class of reconstructed discontinuous Galerkin methods for the compressible flows on arbitrary grids, AIAA-2011-0199, 2011.
[32] H. Luo, H. Xiao, R. Nourgaliev, C. Cai, A comparative study of different reconstruction schemes for reconstructed discontinuous Galerkin methods for the compressible flows on arbitrary grids, AIAA-2011-3839, 2011.
[33] D.F. Haider, J.P. Croisille, B. Courbet, Stability analysis of the cell centered finite-volume MUSCL method on unstructured grids, Numirische Mathematik 113 (4) (2009) 555–600.
[34] D. Balsara, C. Altmann, C.D. Munz, M. Dumbser, A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG + HWENO schemes, Journal of Computational Physics 226 (2007) 586–620.
[35] Z. Xu, Y. Liu, H. Du, S.W. Shu, Point-wise hierarchical reconstruction for discontinuous Galerkin and finite volume method for solving conservation laws, Journal of Computational Physics 230 (2011) 6843–6865.

Journal of Computational Physics 236 (2013) 477–492

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A reconstructed discontinuous Galerkin method based on a Hierarchical WENO reconstruction for compressible flows on tetrahedral grids

Hong Luo [a,*,1], Yidong Xia [a,2], Seth Spiegel [a,2], Robert Nourgaliev [b,3], Zonglin Jiang [c,4]

[a] Department of Mechanical and Aerospace Engineering, North Carolina State University, Raleigh, NC 27695, United States
[b] Idaho National Laboratory, Idaho Falls, ID 83415, United States
[c] Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China

ARTICLE INFO

Article history:
Received 10 August 2012
Received in revised form 22 November 2012
Accepted 27 November 2012
Available online 10 December 2012

Keywords:
Discontinuous Galerkin method
WENO reconstruction
Unstructured grids

ABSTRACT

A reconstructed discontinuous Galerkin (RDG) method based on a hierarchical WENO reconstruction, termed HWENO (P_1P_2) in this paper, designed not only to enhance the accuracy of discontinuous Galerkin methods but also to ensure the nonlinear stability of the RDG method, is presented for solving the compressible Euler equations on tetrahedral grids. In this HWENO (P_1P_2) method, a quadratic polynomial solution (P_2) is first reconstructed using a Hermite WENO reconstruction from the underlying linear polynomial (P_1) discontinuous Galerkin solution to ensure the linear stability of the RDG method and to improve the efficiency of the underlying DG method. By taking advantage of handily available and yet invaluable information, namely the derivatives in the DG formulation, the stencils used in the reconstruction involve only von Neumann neighborhood (adjacent face-neighboring cells) and thus are compact. The first derivatives of the quadratic polynomial solution are then reconstructed using a WENO reconstruction in order to eliminate spurious oscillations in the vicinity of strong discontinuities, thus ensuring the nonlinear stability of the RDG method. The developed HWENO (P_1P_2) method is used to compute a variety of flow problems on tetrahedral meshes to demonstrate its accuracy, robustness, and non-oscillatory property. The numerical experiments indicate that the HWENO (P_1P_2) method is able to capture shock waves within one cell without any spurious oscillations, and achieve the designed third-order of accuracy: one order accuracy higher than the underlying DG method.

1. Introduction

The discontinuous Galerkin methods [1–28] (DGM) have recently become popular for the solution of systems of conservation laws. Originally introduced for the solution of neutron transport equations [1], nowadays they are widely used in computational fluid dynamics, computational acoustics, and computational magneto-hydrodynamics. The discontinuous Galerkin methods combine two advantageous features commonly associated with finite element and finite volume methods.

* Corresponding author.
E-mail address: hong_luo@ncsu.edu (H. Luo).
[1] Associate Professor, Department of Mechanical and Aerospace Engineering.
[2] Ph.D. student, Department of Mechanical and Aerospace Engineering.
[3] Senior Scientist, Thermal Science and Safety Analysis Department.
[4] Professor, State Key Laboratory of High Temperature Gas Dynamics.

0021-9991/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2012.11.026

As in classical finite element methods, accuracy is obtained by means of high-order polynomial approximation within an element rather than by wide stencils as in the case of finite volume methods. The physics of wave propagation is, however, accounted for by solving the Riemann problems that arise from the discontinuous representation of the solution at element interfaces. In this respect, the DG methods are similar to finite volume methods. The discontinuous Galerkin methods have many attractive features: (1) They have several useful mathematical properties with respect to conservation, stability, and convergence; (2) The methods can be easily extended to higher-order (>2nd) approximation; (3) The methods are well suited for complex geometries since they can be applied on unstructured grids. In addition, the methods can also handle non-conforming elements, where the grids are allowed to have hanging nodes; (4) The methods are highly parallelizable, as they are compact and each element is independent. Since the elements are discontinuous, and the inter-element communications are minimal, domain decomposition can be efficiently employed. The compactness also allows for structured and simplified coding for the methods; (5) They can easily handle adaptive strategies, since refining or coarsening a grid can be achieved without considering the continuity restriction commonly associated with the conforming elements. The methods allow easy implementation of *hp*-refinement, for example, the order of accuracy, or shape, can vary from element to element; (6) They have the ability to compute low Mach number flow problems without recourse to the time-preconditioning techniques normally required for the finite volume methods. In contrast to the enormous advances in the theoretical and numerical analysis of the DGM, the development of a viable, attractive, competitive, and ultimately superior DG method over the more mature and well-established second order finite volume methods is relatively an untouched area. This is mainly due to the fact that the DGM have a number of weaknesses that have yet to be addressed, before they can be robustly used for flow problems of practical interest in a complex configuration environment. In particular, how to effectively control spurious oscillations in the presence of strong discontinuities, and how to reduce the computing costs for the DGM remain the two most challenging and unresolved issues in the DGM. Indeed, compared to the finite element methods and finite volume methods, the DGM require solutions of systems of equations with more unknowns for the same grids. Consequently, these methods have been recognized as expensive in terms of both computational costs and storage requirements especially in the context of implicit methods, where the memory requirement for the Jacobian matrix grows quadratically with the order of the DG methods, thus leading to a significant increase in computational cost.

In order to reduce high costs associated with the DGM, Dumbser et al. [18–20] have introduced a new family of reconstructed DGM, termed PnPm schemes and referred to as RDG (P_nP_m) in this paper, where P_n indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and Pm represents a reconstructed polynomial solution of degree of m ($m \geqslant n$) that is used to compute the fluxes. The RDG (P_nP_m) schemes are designed to enhance the accuracy of the discontinuous Galerkin method by increasing the order of the underlying polynomial solution. The beauty of RDG (P_nP_m) schemes is that they provide a unified formulation for both finite volume and DGM, and contain both classical finite volume and standard DG methods as two special cases of RDG (P_nP_m) schemes, and thus allow for a direct efficiency comparison. When n = 0, i.e., a piecewise constant polynomial is used to represent a numerical solution, RDG (P_0P_m) is nothing but classical high order finite volume schemes, where a polynomial solution of degree m ($m \geqslant 1$) is reconstructed from a piecewise constant solution. When m = n, the reconstruction reduces to the identity operator, and RDG (P_nP_n) scheme yields a standard DG method.

Obviously, the construction of an accurate and efficient reconstruction operator is crucial to the success of the RDG (P_nP_m) schemes. In Dumbser's work, a higher order polynomial solution is reconstructed using a L_2 projection, requiring it indistinguishable from the underlying DG solutions in the contributing cells in the weak sense. The resultant over-determined system is then solved using a least-squares method that guarantees exact conservation, not only of the cell averages but also of all higher order moments in the reconstructed cell itself, such as slopes and curvatures. However, this conservative least-squares reconstruction approach is computationally expensive, as the L_2 projection, i.e., the operation of integration, is required to obtain the resulting over-determined system. Furthermore, the reconstruction might be problematic for a boundary cell, where the number of the face-neighboring cells might be not enough to provide the necessary information to recover a polynomial solution of a desired order. Fortunately, the projection-based reconstruction is not the only way to obtain a polynomial solution of higher order from the underlying discontinuous Galerkin solutions. In a reconstructed DG method using a Taylor basis [26–28] developed by Luo et al. for the solution of the compressible Euler and Navier–Stokes equations on arbitrary grids, a higher order polynomial solution is reconstructed by use of a strong interpolation, requiring point values and derivatives to be interpolated on the face-neighboring cells. The resulting over-determined linear system of equations is then solved in the least-squares sense. This reconstruction scheme only involves von Neumann neighborhood, and thus is compact, simple, robust, and flexible. Like the projection-based reconstruction, the strong reconstruction scheme guarantees exact conservation, not only of the cell averages but also of their slopes due to a judicious choice of the Taylor basis. More recently, Zhang et al. [29,30] presented a class of hybrid DG/FV methods for the conservation laws, where the second derivatives in a cell are obtained from the first derivatives in the cell itself and its neighboring cells using a Green-Gauss reconstruction widely used in the finite volume methods. This provides a fast, simple, and robust way to obtain higher-order polynomial solutions. Lately, Luo et al. [31,32] have conducted a comparative study for these three reconstructed discontinuous Galerkin methods RDG (P_1P_2) by solving 2D Euler equations on arbitrary grids. It is found that all three reconstructed discontinuous Galerkin methods can deliver the desired third order of accuracy and significantly improve the accuracy of the underlying second-order DG method, although the least-squares reconstruction method provides the best performance in terms of both accuracy and robustness.

Unfortunately, the attempt to extend our RDG method to solve 3D Euler equations on tetrahedral grids was not successful. Like the second order cell-centered finite volume methods RDG (P_0P_1), the resultant RDG (P_1P_2) method is unstable.

Although RDG (P_0P_1) methods are in general stable in 2D and on Cartesian or structured grids in 3D, they suffer from the so-called *linear instability* on unstructured tetrahedral grids, when the reconstruction stencils only involve von Neumann neighborhood, i.e., adjacent face-neighboring cells [33]. The RDG (P_1P_2) method exhibits the same linear instability, which can be overcome by using extended stencils. However, this is achieved at the expense of sacrificing the compactness of the underlying DG methods. Furthermore, these linear reconstruction-based DG methods will suffer from non-physical oscillations in the vicinity of strong discontinuities for the compressible Euler equations. Alternatively, ENO, WENO, and HWENO can be used to reconstruct a higher-order polynomial solution, thereby not only enhancing the order of accuracy of the underlying DG method but also achieving both linear and non-linear stability. This type of hybrid HWENO + DG schemes has been developed on 1D and 2D structured grids by Balsara et al. [34], where the HWENO reconstruction is relatively simple and straightforward.

The objective of the effort discussed in this paper is to develop a reconstructed discontinuous Galerkin method based on a hierarchical WENO reconstruction, HWENO (P_1P_2), using a Taylor basis [13] for the solution of the compressible Euler equations on unstructured tetrahedral grids. The HWENO (P_1P_2) method is designed not only to reduce the high computing costs of the DGM, but also to avoid spurious oscillations in the vicinity of strong discontinuities, thus effectively addressing the two shortcomings of the DGM. In this HWENO (P_1P_2) method, a quadratic solution is first reconstructed to enhance the accuracy of the underlying DG method in two steps: (1) all second derivatives on each cell are first reconstructed using the solution variables and their first derivatives from adjacent face-neighboring cells via a strong interpolation; (2) the final second derivatives on each cell are then obtained using a WENO strategy based on the reconstructed second derivatives on the cell itself and its adjacent face-neighboring cells. This reconstruction scheme, by taking advantage of handily available and yet valuable information namely the gradients in the context of the DG methods, only involves von Neumann neighborhood and thus is compact, simple, robust, and flexible. As the underlying DG method is second-order, and the basis functions are at most linear functions, fewer quadrature points are then required for both domain and face integrals, and the number of unknowns (the number of degrees of freedom) remains the same as for the DG (P_1). Consequently, this RDG method is more efficient than its third order DG (P_2) counterpart. The gradients of the quadratic polynomial solutions are then modified using a WENO reconstruction in order to eliminate non-physical oscillations in the vicinity of strong discontinuities, thus ensuring the non-linear stability of the RDG method. The developed HWENO (P_1P_2) method is used to compute a variety of flow problems on tetrahedral grids to demonstrate its accuracy, robustness, and non-oscillatory performance. The presented numerical results indicate that this HWENO (P_1P_2) method is able to capture shock waves within once cell without any spurious oscillations, and achieve the designed third-order of accuracy: one order accuracy higher than the underlying DG (P_1) method, and thus significantly increase its accuracy without significant increase in computing costs and memory requirements. The remainder of this paper is organized as follows. The governing equations are listed in Section 2. The developed reconstructed discontinuous Galerkin method is presented in Section 3. Extensive numerical experiments are reported in Section 4. Concluding remarks are given in Section 5.

2. Governing equations

The Euler equations governing unsteady compressible inviscid flows can be expressed as

$$\frac{\partial \mathbf{U}(x,t)}{\partial t}+\frac{\partial \mathbf{F}_k(\mathbf{U}(x,t))}{\partial x_k}=0 \tag{2.1}$$

where the summation convention has been used. The conservative variable vector $\mathbf{U}$, and inviscid flux vector $\mathbf{F}$ are defined by

$$\mathbf{U}=\begin{pmatrix}\rho\\ \rho u_i\\ \rho e\end{pmatrix}\quad \mathbf{F}_j=\begin{pmatrix}\rho u_j\\ \rho u_i u_j+p\delta_{ij}\\ u_j(\rho e+p)\end{pmatrix} \tag{2.2}$$

Here ρ, p, and e denote the density, pressure, and specific total energy of the fluid, respectively, and u_i is the velocity of the flow in the coordinate direction x_i. The pressure can be computed from the equation of state

$$p=(\gamma-1)\rho\left(e-\frac{1}{2}u_j u_j\right) \tag{2.3}$$

which is valid for perfect gas, where γ is the ratio of the specific heats.

3. Reconstructed discontinuous Galerkin method

3.1. Discontinuous Galerkin formulation

The governing Eq. (2.1) is discretized using a discontinuous Galerkin finite element formulation. We first introduce some notations. We assume that the domain Ω is subdivided into a collection of non-overlapping tetrahedral elements Ω_e. We introduce the following broken Sobolev space $\boldsymbol{V}_h^p$

$$V_h^p = \left\{ v_h \in [L_2(\Omega)]^m : v_h|_{\Omega_e} \in \left[V_p^m\right] \forall \Omega_e \in \Omega \right\}, \tag{3.1}$$

which consists of discontinuous vector-values polynomial functions of degree p, and where m is the dimension of the unknown vector and V_p is the space of all polynomials of degree $\leqslant p$. To formulate the discontinuous Galerkin method, we introduce the following weak formulation, which is obtained by multiplying the above conservation law (2.1) by a test function $\mathbf{W}_h$, integrating over an element Ω_e, and then performing an integration by parts,

Find $\mathbf{U}_h \in V_h^p$ such as

$$\frac{d}{dt}\int_{\Omega_e} \mathbf{U}_h \mathbf{W}_h d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k \mathbf{W}_h d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h)\frac{\partial \mathbf{W}_h}{\partial x_k} d\Omega = 0, \quad \forall \mathbf{W}_h \in V_h^p \tag{3.2}$$

where $\mathbf{U}_h$ and $\mathbf{W}_h$ are represented by piecewise-polynomial functions of degrees p, which are discontinuous between the cell interfaces and $\mathbf{n}_k$ denotes the unit outward normal vector to Γ_e: the boundary of Ω_e. Assume that B_i is the basis of polynomial function of degrees p, this is then equivalent to the following system of N equations,

$$\frac{d}{dt}\int_{\Omega_e} \mathbf{U}_h B_i d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k B_i d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h)\frac{\partial B_i}{\partial x_k} d\Omega = 0, \quad 1 \leqslant i \leqslant \mathrm{N} \tag{3.3}$$

where N is the dimension of the polynomial space. Since the numerical solution $\mathbf{U}_h$ is discontinuous between element interfaces, the interface fluxes are not uniquely defined. The flux function $\mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k$ appearing in the second terms of Eq. (3.3) is replaced by a numerical Riemann flux function $\mathbf{H}_k\left(\mathbf{U}_h^L, \mathbf{U}_h^R, \mathbf{n}_k\right)$ where $\mathbf{U}_h^L$ and $\mathbf{U}_h^R$ are the conservative state vector at the left and right side of the element boundary, and which is computed using HLLC scheme [43] in the present work. This scheme is called discontinuous Galerkin method of degree p, or in short notation DG (P) method. Note that discontinuous Galerkin formulations are very similar to finite volume schemes, especially in their use of numerical fluxes. Indeed, the classical first-order cell-centered finite volume scheme exactly corresponds to the DG (P_0) method, i.e., to the discontinuous Galerkin method using a piecewise-constant polynomial. Consequently, the DG (P_k) methods with $k > 0$ can be regarded as a natural generalization of finite volume methods to higher order methods. By simply increasing the degree p of the polynomials, the DG methods of corresponding higher order are obtained. The domain and boundary integrals in Eq. (3.3) are calculated using Gauss quadrature formulas. The number of quadrature points used is chosen to integrate exactly polynomials of order of $2p$ and $2p + 1$ for volume and surface inner products in the reference element. In the traditional DGM, termed nodal discontinuous Galerkin methods, numerical polynomial solutions $\mathbf{U}_h$ in each element are expressed using either standard Lagrange finite element or hierarchical node-based basis as following

$$\mathbf{U}_h = \sum_{i=1}^{\mathrm{N}} \mathbf{U}_i(t) B_i(x), \tag{3.4}$$

where B_i are the finite element basis functions. As a result, the unknowns to be solved for are the variables at the nodes $\mathbf{U}_i$.

In the present work, the numerical polynomial solutions are represented using a Taylor series expansion at the center of the cell. For example, if we do a Taylor series expansion at the center of the cell, a quadratic polynomial solution can be expressed as follows:

$$\begin{aligned}\mathbf{U}_h = \mathbf{U}_c &+ \left.\frac{\partial \mathbf{U}}{\partial x}\right|_c (x - x_c) + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c (y - y_c) + \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c (z - z_c) + \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c \frac{(x - x_c)^2}{2} + \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c \frac{(y - y_c)^2}{2} + \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c \frac{(z - z_c)^2}{2} \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial y}\right|_c (x - x_c)(y - y_c) + \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial z}\right|_c (x - x_c)(z - z_c) + \left.\frac{\partial^2 \mathbf{U}}{\partial y \partial z}\right|_c (y - y_c)(z - z_c)\end{aligned} \tag{3.5}$$

which can be further expressed as cell-averaged values and their derivatives at the center of the cell:

$$\begin{aligned}\mathbf{U}_h = \widetilde{\mathbf{U}} &+ \left.\frac{\partial \mathbf{U}}{\partial x}\right|_c (x - x_c) + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c (y - y_c) + \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c (z - z_c) + \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c \left(\frac{(x - x_c)^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{(x - x_c)^2}{2} d\Omega\right) \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c \left(\frac{(y - y_c)^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{(y - y_c)^2}{2} d\Omega\right) + \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c \left(\frac{(z - z_c)^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{(z - z_c)^2}{2} d\Omega\right) \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial y}\right|_c \left((x - x_c)(y - y_c) - \frac{1}{\Omega_e}\int_{\Omega_e} (x - x_c)(y - y_c) d\Omega\right) \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial z}\right|_c \left((x - x_c)(z - z_c) - \frac{1}{\Omega_e}\int_{\Omega_e} (x - x_c)(z - z_c) d\Omega\right) \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial y \partial z}\right|_c \left((y - y_c)(z - z_c) - \frac{1}{\Omega_e}\int_{\Omega_e} (y - y_c)(z - z_c) d\Omega\right)\end{aligned} \tag{3.6}$$

where $\widetilde{\mathbf{U}}$ is the mean value of $\mathbf{U}$ in this cell. The unknowns to be solved for in this formulation are the cell-averaged variables and their derivatives at the center of the cells. The dimension of the polynomial space is 10 and the basis functions are

$$
\begin{aligned}
&B_1 = 1, \quad B_2 = x - x_c, \quad B_3 = y - y_c, \quad B_4 = z - z_c \\
&B_5 = \frac{B_2^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_2^2}{2}d\Omega, \quad B_6 = \frac{B_3^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_3^2}{2}d\Omega, \quad B_7 = \frac{B_4^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_4^2}{2}d\Omega, \\
&B_8 = B_2B_3 - \frac{1}{\Omega_e}\int_{\Omega_e}B_2B_3d\Omega, \quad B_9 = B_2B_4 - \frac{1}{\Omega_e}\int_{\Omega_e}B_2B_4d\Omega, \quad B_{10} = B_3B_4 - \frac{1}{\Omega_e}\int_{\Omega_e}B_3B_4d\Omega
\end{aligned} \tag{3.7}
$$

The discontinuous Galerkin formulation then leads to the following 10 equations

$$
\begin{aligned}
&\frac{d}{dt}\int_{\Omega_e}\widetilde{\mathbf{U}}d\Omega + \int_{\Gamma_e}H_k\left(\mathbf{U}_h^L, \mathbf{U}_h^R, \mathbf{n}_k\right)d\Gamma = 0 \\
&M_{9\times 9}\frac{d}{dt}\left(\left.\frac{\partial \mathbf{U}}{\partial x}\right|_c \quad \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c \quad \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c \quad \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c \quad \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c \quad \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c \quad \left.\frac{\partial^2 \mathbf{U}}{\partial x\partial y}\right|_c \quad \left.\frac{\partial^2 \mathbf{U}}{\partial x\partial z}\right|_c \quad \left.\frac{\partial^2 \mathbf{U}}{\partial y\partial z}\right|_c\right)^T + R_{9\times 1} = 0
\end{aligned} \tag{3.8}
$$

Note that in this formulation, equations for the cell-averaged variables are decoupled from equations for their derivatives due to the judicial choice of the basis functions and the fact that

$$
\int_{\Omega_e} B_1B_i d\Omega = 0 \quad 2 \leqslant i \leqslant 10 \tag{3.9}
$$

In the implementation of this DG method, the basis functions are actually normalized in order to improve the conditioning of the system matrix (3.3) as follows:

$$
\begin{aligned}
&B_1 = 1, \quad B_2 = \frac{x - x_c}{\Delta x}, \quad B_3 = \frac{y - y_c}{\Delta y}, \quad B_4 = \frac{z - z_c}{\Delta z} \\
&B_5 = \frac{B_2^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_2^2}{2}d\Omega, \quad B_6 = \frac{B_3^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_3^2}{2}d\Omega, \quad B_7 = \frac{B_4^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_4^2}{2}d\Omega, \\
&B_8 = B_2B_3 - \frac{1}{\Omega_e}\int_{\Omega_e}B_2B_3d\Omega, \quad B_9 = B_2B_4 - \frac{1}{\Omega_e}\int_{\Omega_e}B_2B_4d\Omega, \quad B_{10} = B_3B_4 - \frac{1}{\Omega_e}\int_{\Omega_e}B_3B_4d\Omega,
\end{aligned} \tag{3.10}
$$

where $\Delta x = 0.5(x_{\max} - x_{\min})$, $\Delta y = 0.5(y_{\max} - y_{\min})$, and $\Delta z = 0.5(z_{\max} - z_{\min})$, and $x_{\max}$, $x_{\min}$, $y_{\max}$, $y_{\min}$, $z_{\max}$, and $z_{\min}$ are the maximum and minimum coordinates in the cell Ω_e in x-, y-, and z-directions, respectively. A quadratic polynomial solution can then be rewritten as

$$
\mathbf{U}_h = \widetilde{\mathbf{U}}B_1 + \mathbf{U}_xB_2 + \mathbf{U}_yB_3 + \mathbf{U}_zB_4 + \mathbf{U}_{xx}B_5 + \mathbf{U}_{yy}B_6 + \mathbf{U}_{zz}B_7 + \mathbf{U}_{xy}B_8 + \mathbf{U}_{xz}B_9 + \mathbf{U}_{yz}B_{10} \tag{3.11}
$$

where

$$
\begin{aligned}
&\mathbf{U}_{\mathbf{x}} = \left.\frac{\partial \mathbf{U}}{\partial x}\right|_c \Delta x, \quad \mathbf{U}_y = \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c \Delta y, \quad \mathbf{U}_z = \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c \Delta z, \quad \mathbf{U}_{xx} = \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c \Delta x^2, \quad \mathbf{U}_{yy} = \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c \Delta y^2, \\
&\mathbf{U}_{zz} = \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c \Delta z^2, \quad \mathbf{U}_{xy} = \left.\frac{\partial^2 \mathbf{U}}{\partial x\partial y}\right|_c \Delta x\Delta y, \quad \mathbf{U}_{xz} = \left.\frac{\partial^2 \mathbf{U}}{\partial x\partial z}\right|_c \Delta x\Delta z, \quad \mathbf{U}_{yz} = \left.\frac{\partial^2 \mathbf{U}}{\partial y\partial z}\right|_c \Delta y\Delta z
\end{aligned} \tag{3.12}
$$

This formulation belongs to the so-called modal discontinuous Galerkin method and has a number of attractive, distinct, and useful features. First, cell-averaged variables and their derivatives are handily available in this formulation. This makes the implementation of both in-cell and inter-cell reconstruction schemes straightforward and simple [26–28,30,31]. Secondly, the Taylor basis is hierarchic. This greatly facilitates implementation of p-multigrid methods [16,17] and p-refinement. Thirdly, the same basis functions are used for any shapes of elements: tetrahedron, pyramid, prism, and hexahedron. This makes the implementation of DGM on arbitrary meshes straightforward.

3.2. Reconstructed discontinuous Galerkin methods

In comparison with reconstructed FV methods, the DGM have a significant drawback in that they require more degrees of freedom, additional domain integration, and more Gauss quadrature points for the boundary integration, and therefore more computational costs and storage requirements. On the one hand, the reconstruction methods that FV methods use to achieve higher-order accuracy are relatively inexpensive but less accurate and robust. On the other hand, the DGM that can be viewed as a different way to extend a FV method to higher orders are accurate and robust but costly. It is only natural and tempting to combine the efficiency of the reconstruction methods and the accuracy of the DG methods. This idea was originally introduced by Dumbser et al. in the frame of P_nP_m scheme [18–20], termed RDG (P_nP_m) in this paper, where Pn indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and Pm represents a reconstructed polynomial solution of degree of m ($m \geqslant n$) that is used to compute the fluxes and source terms. The beauty of RDG (P_nP_m) schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of RDG (P_nP_m) schemes, and thus allow for a direct efficiency

comparison. When n = 0, i.e., a piecewise constant polynomial is used to represent a numerical solution, RDG (P_0P_m) is nothing but classical high order finite volume schemes, where a polynomial solution of degree m ($m \geqslant 1$) is reconstructed from a piecewise constant solution. When m = n, the reconstruction reduces to the identity operator, and RDG (P_nP_m) scheme yields a standard DG method. Clearly, an accurate and efficient reconstruction is the key ingredient in extending the underlying DG method to higher order accuracy. Our discussion in this work is mainly focused on a third order RDG (P_1P_2) method, as the benefits of higher-order (>3rd) methods diminish dramatically for aerodynamic applications. Nevertheless, its extension to higher order DG methods is straightforward, as demonstrated by Dumbser et al. in Ref. [18]. The RDG (P_1P_2) method is based on a hierarchical Hermite WENO reconstruction and designed not only to reduce the high computing costs of the DGM, but also to avoid spurious oscillations in the vicinity of strong discontinuities, thus ensuring the nonlinear stability of the RDG method. Similar to moment limiters, the hierarchical reconstruction methods [35] reconstruct the derivatives in a hierarchical manner. In the case of the RDG (P_1P_2) method, the second derivatives (curvatures) are first reconstructed and the first derivatives (gradients) are then reconstructed, which are describes in the next two sub-sections.

3.2.1. WENO reconstruction at P_2: WENO (P_1P_2)

The reconstruction of the second derivatives consists of two steps: a quadratic polynomial solution (P_2) is first reconstructed using a least-squares method from the underlying linear polynomial (P_1) discontinuous Galerkin solution, and the final quadratic polynomial solution is then obtained using a WENO reconstruction, which is necessary to ensure the linear stability of the RDG method [36]. The resulting RDG method is referred to as WENO (P_1P_2) in this paper, where the quadratic polynomial solution is obtained from the underlying linear DG solution via a WENO reconstruction.

3.2.1.1. Least-squares reconstruction. In the case of DG (P_1) method, a linear polynomial solution $\mathbf{U}_i$ in any cell i is expressed as

$$\mathbf{U}_i = \widetilde{\mathbf{U}}_i + \mathbf{U}_{\mathbf{x}i}B_2 + \mathbf{U}_{\mathbf{y}i}B_3 + \mathbf{U}_{zi}B_4 \tag{3.13}$$

A quadratic polynomial solution $\mathbf{U}_i^R$ can be reconstructed using the underlying linear polynomial DG solution in the neighboring cells as follows:

$$\mathbf{U}_i^R = \widetilde{\mathbf{U}}_i^R + \mathbf{U}_{xi}^R B_2 + \mathbf{U}_{yi}^R B_3 + \mathbf{U}_{zi}^R B_4 + \mathbf{U}_{xxi}^R B_5 + \mathbf{U}_{yyi}^R B_6 + \mathbf{U}_{zzi}^R B_7 + \mathbf{U}_{xyi}^R B_8 + \mathbf{U}_{xzi}^R B_9 + \mathbf{U}_{yzi}^R B_{10} \tag{3.14}$$

In order to maintain the compactness of the DG methods, the reconstruction is required to involve only von Neumann neighborhood, i.e., the adjacent cells that share a face with the cell i under consideration. There are 10 degrees of freedom, and therefore 10 unknowns must be determined. The first four unknowns can be trivially obtained, by requiring the consistency of the RDG with the underlying DG: (1) The reconstruction scheme must be conservative, and (2) The values of the reconstructed first derivatives are equal to the ones of the first derivatives of the underlying DG solution at the centroid i. Due to the judicious choice of Taylor basis in our DG formulation, these four degrees of freedom simply coincide with the ones from the underlying DG solution, i.e.,

$$\widetilde{\mathbf{U}}_i^R = \widetilde{\mathbf{U}}_i, \mathbf{U}_{xi}^R = \mathbf{U}_{xi}, \mathbf{U}_{yi}^R = \mathbf{U}_{yi}, \mathbf{U}_{zi}^R = \mathbf{U}_{zi} \tag{3.15}$$

As a result, only six second derivatives need to be determined. This can be accomplished by requiring that the point-wise values and first derivatives of the reconstructed solution are equal to these of the underlying DG solution at the cell centers for all the adjacent face neighboring cells. Considering an adjacent neighboring cell j, one obtains

$$\begin{aligned}
&\mathbf{U}_j = \widetilde{\mathbf{U}}_i + \mathbf{U}_{xi}B_2 + \mathbf{U}_{yi}B_3 + \mathbf{U}_{zi}B_4 + \mathbf{U}_{xxi}^R B_5 + \mathbf{U}_{yyi}^R B_6 + \mathbf{U}_{zzi}^R B_7 + \mathbf{U}_{xyi}^R B_8 + \mathbf{U}_{xzi}^R B_9 + \mathbf{U}_{yzi}^R B_{10} \\
&\left.\frac{\partial \mathbf{U}}{\partial x}\right|_j = \mathbf{U}_{xi}\frac{1}{\Delta x_i} + \mathbf{U}_{xxi}^R\frac{B_2}{\Delta x_i} + \mathbf{U}_{xyi}^R\frac{B_3}{\Delta x_i} + \mathbf{U}_{xzi}^R\frac{B_4}{\Delta x_i} \\
&\left.\frac{\partial \mathbf{U}}{\partial y}\right|_j = \mathbf{U}_{yi}\frac{1}{\Delta y_i} + \mathbf{U}_{yyi}^R\frac{B_3}{\Delta y_i} + \mathbf{U}_{xyi}^R\frac{B_2}{\Delta y_i} + \mathbf{U}_{yzi}^R\frac{B_4}{\Delta y_i} \\
&\left.\frac{\partial \mathbf{U}}{\partial z}\right|_j = \mathbf{U}_{zi}\frac{1}{\Delta z_i} + \mathbf{U}_{zzi}^R\frac{B_4}{\Delta z_i} + \mathbf{U}_{xzi}^R\frac{B_2}{\Delta z_i} + \mathbf{U}_{yzi}^R\frac{B_3}{\Delta z_i}
\end{aligned} \tag{3.16}$$

where the basis functions B are evaluated at the center of cell j, i.e., $B = \mathbf{B}(x_j,y_j,z_j)$. This can be written in a matrix form as follows:

$$\begin{pmatrix} B_5^j & B_6^j & B_7^j & B_8^j & B_9^j & B_{10}^j \\ B_2^j & 0 & 0 & B_3^j & B_4^j & 0 \\ 0 & B_3^j & 0 & B_2^j & 0 & B_4^j \\ 0 & 0 & B_4^j & 0 & B_2^j & B_3^j \end{pmatrix} \begin{pmatrix} \mathbf{U}_{xxi}^R \\ \mathbf{U}_{yyi}^R \\ \mathbf{U}_{zzi}^R \\ \mathbf{U}_{xyi}^R \\ \mathbf{U}_{xzi}^R \\ \mathbf{U}_{yzi}^R \end{pmatrix} = \begin{pmatrix} \mathbf{U}_j - \left(\mathbf{U}_i B_1^j + \mathbf{U}_{xi}B_2^j + \mathbf{U}_{yi}B_3^j + \mathbf{U}_{zi}B_4^j\right) \\ \frac{\Delta x_i}{\Delta x_j}\mathbf{U}_{xj} - \mathbf{U}_{xi} \\ \frac{\Delta y_i}{\Delta y_j}\mathbf{U}_{yj} - \mathbf{U}_{yi} \\ \frac{\Delta z_i}{\Delta z_j}\mathbf{U}_{zj} - \mathbf{U}_{zi} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1^j \\ \mathbf{R}_2^j \\ \mathbf{R}_3^j \\ \mathbf{R}_4^j \end{pmatrix} \tag{3.17}$$

where **R** is used to represent the right-hand-side for simplicity. Similar equations can be written for all cells connected to the cell i with a common face, which leads to a non-square matrix. The number of face-neighboring cells for a tetrahedral is four. Consequently, the size of the resulting non-square matrix is 16×6. In the present work, this over-determined linear system of 16 equations for 6 unknowns is solved in the least-squares sense using both normal equation approach and the QR decomposition to obtain the second derivatives of the reconstructed quadratic polynomial solution. One can easily verify that this least-squares reconstruction satisfies the so-called 2-exactness, i.e., it can reconstruct a quadratic polynomial function exactly.

3.2.1.2. WENO reconstruction. This least-squares reconstructed discontinuous Galerkin method: RDG (P_1P_2) has been successfully used to solve the 2D compressible Euler equations for smooth flows on arbitrary grids [26–28,30,31] and is able to achieve the designed third order of accuracy and significantly improve the accuracy of the underlying second-order DG method. However, when extended to solve the 3D compressible Euler equations on tetrahedral grids, this RDG method suffers from the so-called linear instability, that is also observed in the second-order cell-centered finite volume methods, i.e., RDG (P_0P_1)[33]. This linear instability is attributed to the fact that the reconstruction stencils only involve von Neumann neighborhood, i.e., adjacent face-neighboring cells [33]. The linear stability can be achieved using extended stencils, which will unfortunately sacrifice the compactness of the underlying DG methods. Furthermore, such a linear reconstruction-based DG method cannot maintain the non-linear instability, leading to non-physical oscillations in the vicinity of strong discontinuities. Alternatively, ENO/WENO can be used to reconstruct a higher-order polynomial solution, which can not only enhance the order of accuracy of the underlying DG method but also achieve both linear and non-linear stability. Specifically, the WENO scheme introduced by Dumber et al. [40] is adopted in this work, where an entire quadratic polynomial solution on cell i is obtained using a nonlinear WENO reconstruction as a convex combination of the least-squares reconstructed second derivatives at the cell itself and its four face-neighboring cells,

$$\left.\frac{\partial^2 \mathbf{U}}{\partial x_m \partial x_n}\right|_i^{\mathrm{WENO}} = \sum_{k=1}^{5} w_k \left.\frac{\partial^2 \mathbf{U}}{\partial x_m \partial x_n}\right|_k \tag{3.18}$$

where the normalized nonlinear weights w_k are computed as

$$w_k = \frac{\tilde{w}_k}{\sum_{i=1}^{5} \tilde{w}_k} \tag{3.19}$$

The non-normalized nonlinear weights $\tilde{w}_i$ are functions of the linear weights λ_i and the so-called oscillation indicator o_i

$$\tilde{w}_i = \frac{\lambda_i}{(\varepsilon + o_i)^{\gamma}} \tag{3.20}$$

where ε is a small positive number used to avoid division by zero, and γ an integer parameter to control how fast the nonlinear weights decay for non-smooth stencils. The oscillation indicator for the reconstructed second order polynomials is simply defined as

$$o_k = \left[\left(\left.\frac{\partial^2 U}{\partial x_m \partial x_n}\right|_k \right)^2 \right]^{1/2} \tag{3.21}$$

The linear weights λ_i can be chosen to balance the accuracy and the non-oscillatory property of the RDG method. Note that the least-squares reconstructed polynomial at the cell itself serves as the central stencil and the least-squares reconstructed polynomials on its four face-neighboring cells act as biased stencils in this WENO reconstruction. This reconstructed quadratic polynomial solution is then used to compute the domain and boundary integrals of the underlying DG (P_1) method in Eq. (3.3). As demonstrated in Ref. [36], the resulting WENO (P_1P_2) method is able to achieve the designed third order of accuracy, maintain the linear stability, and significantly improve the accuracy of the underlying second-order DG method without significant increase in computing costs and storage requirements. Note that this RDG method is not compact anymore, as neighbor's neighbors are used in the solution update. However, the stencil used in the reconstruction is compact, involving only von Neumann neighbors. Consequently, the resultant RDG method can be implemented in a compact manner.

3.2.2. WENO reconstruction at P_1: HWENO (P_1P_2)

Although the WENO (P_1P_2) method does not introduce any new oscillatory behavior for the reconstructed curvature terms (second derivatives) due to the WENO reconstruction, it cannot remove inherent oscillations in the underlying DG (P_1) solutions. Consequently, the WENO (P_1P_2) method still suffers from the non-linear instability for flows with strong discontinuities. In order to eliminate non-physical oscillations in the vicinity of strong discontinuities and thus maintain the non-linear instability, the first derivatives need to be reconstructed using a WENO reconstruction. The resulting reconstructed discontinuous Galerkin method based on this Hierarchical WENO reconstruction is termed as HWENO (P_1P_2) in this paper, where a hierarchical reconstruction (successively from high order to low order) strategy [35] is adopted.

The WENO reconstruction for the first derivatives is based on the reconstructed quadratic polynomial solutions of the flow variables for each cell in the mesh. The stencils are only chosen in the von Neumann neighborhood. More precisely, for a cell i, the following four stencils $(i,j1,j2,j3),(i,j1,j2,j4)$, i.e., $j1,j3,j4$), and $(i,j2,j3,j4)$, where $j1,j2,j3$, and $j4$ designate the four adjacent face-neighboring cells of the cell i are chosen to construct a Lagrange polynomial such that

$$\mathbf{U}_j = \widetilde{\mathbf{U}}_i + \mathbf{U}^R_{xi}B_2 + \mathbf{U}^R_{yi}B_3 + \mathbf{U}^R_{zi}B_4 + \mathbf{U}^R_{xxi}B_5 + \mathbf{U}^R_{yyi}B_6 + \mathbf{U}^R_{zzi}B_7 + \mathbf{U}^R_{xyi}B_8 + \mathbf{U}^R_{xzi}B_9 + \mathbf{U}^R_{yzi}B_{10} \tag{3.22}$$

where $\mathbf{U}_j$ refers to the pointwise value of the reconstructed polynomial solution at centroid of cell j and the basis functions B are evaluated at the center of cell j, i.e., $B = \mathbf{B}(x_j, y_j, z_j)$. In addition, the following four stencils $(i,j1),(i,j2),(i,j3)$, and $(i,j4)$ are chosen to construct a Hermite polynomial such that

$$\begin{aligned}
\left.\frac{\partial \mathbf{U}}{\partial x}\right|_j &= \mathbf{U}^R_{xi}\frac{1}{\Delta x_i} + \mathbf{U}^R_{xxi}\frac{B_2}{\Delta x_i} + \mathbf{U}^R_{xyi}\frac{B_3}{\Delta x_i} + \mathbf{U}^R_{xzi}\frac{B_4}{\Delta x_i} \\
\left.\frac{\partial \mathbf{U}}{\partial y}\right|_j &= \mathbf{U}^R_{yi}\frac{1}{\Delta y_i} + \mathbf{U}^R_{yyi}\frac{B_3}{\Delta y_i} + \mathbf{U}^R_{xyi}\frac{B_2}{\Delta y_i} + \mathbf{U}^R_{yzi}\frac{B_4}{\Delta y_i} \\
\left.\frac{\partial \mathbf{U}}{\partial z}\right|_j &= \mathbf{U}^R_{zi}\frac{1}{\Delta z_i} + \mathbf{U}^R_{zzi}\frac{B_4}{\Delta z_i} + \mathbf{U}^R_{xzi}\frac{B_2}{\Delta z_i} + \mathbf{U}^R_{yzi}\frac{B_3}{\Delta z_i}
\end{aligned} \tag{3.23}$$

These eight reconstructed gradients $\left(\mathbf{U}^R_{xi}, \mathbf{U}^R_{yi},\right.$ and $\left.\mathbf{U}^R_{zi}\right)$ serving as the biased stencils and the gradient from the DG solution itself at cell i ($\mathbf{U}_{xi}$, $\mathbf{U}_{yi}$, and $\mathbf{U}_{zi}$) acting as the central stencil are used to modify the first derivatives based on the WENO reconstruction as a convex combination of these nine derivatives,

$$\left.\frac{\partial \mathbf{U}}{\partial x_m}\right|_i^{\mathrm{WENO}} = \sum_{k=1}^{9} w_k \left.\frac{\partial \mathbf{U}}{\partial x_m}\right|_k \tag{3.24}$$

where the normalized nonlinear weights w_k are computed as

$$w_k = \frac{\tilde{w}_k}{\sum_{i=1}^{9}\tilde{w}_k} \tag{3.25}$$

The non-normalized nonlinear weights $\tilde{w}_i$ are functions of the linear weights λ_i and the so-called oscillation indicator o_i

$$\tilde{w}_i = \frac{\lambda_i}{(\varepsilon + o_i)^{\gamma}} \tag{3.26}$$

where ε is a small positive number used to avoid division by zero, and γ an integer parameter to control how fast the nonlinear weights decay for non-smooth stencils. The oscillation indicator is simply defined as

$$o_k = \left[\left(\left.\frac{\partial \mathbf{U}}{\partial x_m}\right|_k\right)^2\right]^{1/2} \tag{3.27}$$

The present choice of stencils is symmetric, and compact, as van Neumann neighbors are only involved in the reconstruction. This means that no additional data structure is required for our HWENO (P_1P_2) method. Note that this WENO reconstruction at P_1 is the extension of a HWENO limiter developed for the DG (P_1) by the authors in Ref. [15]. From the perspective of both computational cost and solution accuracy, the above WENO reconstruction on P_1 should only be used in the regions where strong discontinuities exist. This can be accomplished using the so-called discontinuity detectors, which are helpful to distinguish regions where solutions are smooth and discontinuous. The beauty of this WENO reconstruction is that in case that the reconstruction is mistakenly applied in the smooth cells, the uniform high-order accuracy can still be maintained, unlike the slope limiters, which, when applied near smooth extrema, will have a profoundly adverse impact on solution in the smooth region, leading to the loss of the original high-order accuracy. This remarkable feature of the WENO reconstruction in turn alleviates the burden on the discontinuity detectors, as no discontinuity detectors can really either in theory or in practice make a distinction between a stagnation point and a shock wave, as flow gradients near the stagnation point are even larger than the ones near the shock wave in some cases. All numerical experiments presented in the next section are performed by applying the P_1 reconstruction everywhere in an effort to ensure that the computational results are not affected by a shock detector, and to demonstrate the superior properties of the designed HWENO (P_1P_2) method.

4. Numerical examples

The hierarchical WENO reconstruction method has been implemented in a well-tested DG code [13–17] to solve a variety of the compressible flow problems on tetrahedral grids, where a fast implicit p-multigrid method [16,17] is used to obtain steady state solutions considered in this paper. A few examples are presented in this section to demonstrate that the developed HWENO (P_1P_2) method is able to maintain the non-linear stability and achieve the designed third order of accuracy. The first two test cases are chosen to demonstrate that the developed HWENO (P_1P_2) method is able to achieve the designed third-order of convergence for smooth flows. The next two test cases are used to assess the non-oscillatory property of

the HWENO (P_1P_2) method for flows with strong discontinuities. The final test case is presented to illustrate the applicability of the HWENO (P_1P_2) method to solve problems of scientific and industrial interests for complex configurations. All the grids in our numerical experiments are generated using an advancing front method described in Refs. [41,42]. In the grid convergence study of the first two test cases, a sequence of the three successively refined tetrahedral grids is used, where the cell size is halved between consecutive meshes. The number of elements on a successively refined mesh is not exactly eight times the coarse mesh's elements due to the nature of unstructured grid generation. The length scale, characterizing the cell size of an unstructured grid, is defined as $1/\sqrt[3]{nDOFs}$, where *nDOFs* is the total number of degrees of freedom. Since the analytical solutions are unknown, the following L^2-norm of the entropy production is used as the error measurement

$$\|\varepsilon\|_{L_2(\Omega)} = \sqrt{\int_\Omega \varepsilon^2 d\Omega}$$

where the entropy production ε defined as

$$\varepsilon = \frac{S - S_\infty}{S_\infty} = \frac{p}{p_\infty}\left(\frac{\rho_\infty}{\rho}\right)^\gamma - 1$$

Note that the entropy production, where the entropy is defined as $S = p/\rho^\gamma$, is a very good criterion to measure accuracy of the numerical solutions, since the flow under consideration is smooth and therefore isentropic.

4.1. Subsonic flow through a channel with a smooth bump

The problem under consideration is a subsonic internal flow inside a 3D channel with a smooth bump on the lower surface. The height, width, and length of the channel are 0.8, 0.8, and 3, respectively. The shape of the lower wall is defined by the function $0.0625\exp(-25 \times {}^2)$ from $x = -1.5$ to $x = 1.5$. The inflow condition is prescribed at a Mach number of 0.5, and an angle of attack of 0°. Fig. 1 shows the three successively refined tetrahedral grids used in this the grid convergence study. The numbers of elements, points, and boundary points for the coarse, medium, and fine grids are (889,254,171), (6986,1555,691), and (55703,10822,2711), respectively. Numerical solutions to this problem are computed using the RDG (P_1P_1), WENO (P_1P_2), and HWENO (P_1P_2) methods on these three grids to obtain a quantitative measurement of the order of accuracy and discretization errors. Fig. 2 illustrates the computed velocity contours in the flow field obtained by the HWENO (P_1P_2) methods on these three grids. The errors and the convergence rates for the three methods are reported in Table 1. It shows the mesh size, the L^2-error of the error function, and the order of convergence. Considering that this is a 3D simulation of a 2D problem, and unstructured tetrahedral grids are not symmetric by nature, thus causing error in the z-direction, the second order RDG (P_1P_1) method can be viewed to offer the designed second order of accuracy. Both WENO (P_1P_2) and HWENO (P_1P_2) methods are able to deliver the designed third order of convergence, adding one order of accuracy to the underlying DG (P_1) method. As expected, the WENO reconstruction on P_1 increases slightly the absolute error. However, it

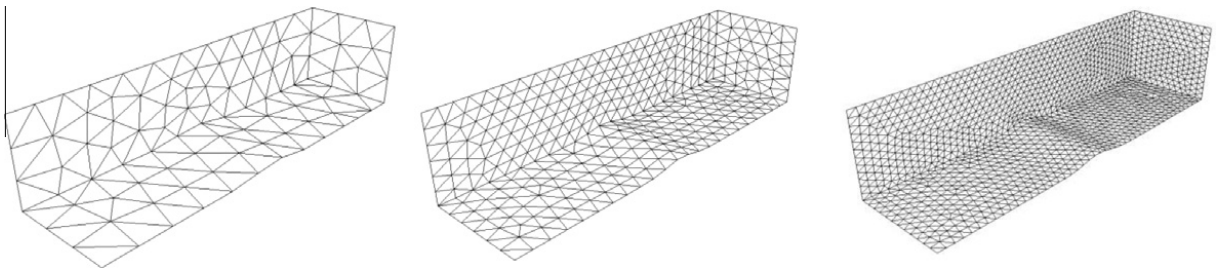

Fig. 1. A sequence of three successively globally refined unstructured meshes used for computing a subsonic flow in a channel with a smooth bump.

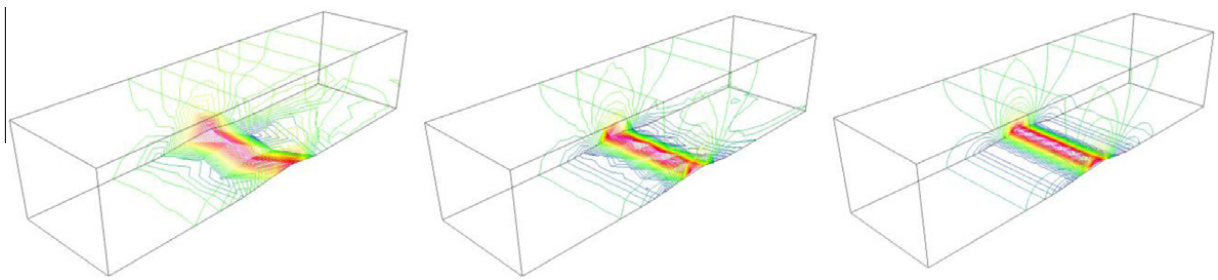

Fig. 2. Computed velocity contours in the flow field obtained by the HWENO (P_1P_2) on a sequence of three successively globally refined unstructured grids for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$.

Table 1
L^2-error and order of convergence for the RDG (P_1P_1), WENO (P_1P_2), and HWENO (P_1P_2) methods.

Length scale	RDG (P_1P_1)		WENO (P_1P_2)		HWENO (P_1P_2)	
	L^2-error	Order	L^2-error	Order	L^2-error	Order
6.552E−2	2.438E−3		2.183E−3		2.220E−3	
3.295E−2	7.356E−4	1.744	2.794E−4	2.992	2.851E−4	2.987
1.650E−2	1.807e-4	2.032	4.539E−05	2.626	4.565E−5	2.647

does not destroy the order of accuracy. This example demonstrates that the HWENO (P_1P_2) method can indeed achieve a third-order rate of convergence for this smooth internal flow.

4.2. Subsonic flow past a sphere

A subsonic flow past a sphere at a Mach number of $M_\infty = 0.5$ is considered in this test case. A sequence of the three successively refined tetrahedral grids used in this grid convergence study is shown in Fig. 3. The numbers of elements, points, and boundary points for the coarse, medium, and fine grids are (535,167,124), (2426,598,322), and (16467,3425,1188), respectively. Like the previous test case, the computations are performed on these three grids using the RDG (P_1P_1), WENO (P_1P_2), and HWENO (P_1P_2) methods. Fig. 4 illustrates the computed velocity contours in the flow field obtained by the HWENO (P_1P_2) method on these three grids. The errors and the orders of accuracy for the three methods are reported in Table 2. All three methods achieve higher than the expected rates of convergence, being 2.36, 3.55, and 3.50 respectively. One can observe again that the HWENO (P_1P_2) keeps the designed order of accuracy, although the WENO reconstruction on P_1 increases slightly the absolute errors of the WENO (P_1P_2) method. This example demonstrates that the HWENO (P_1P_2) method is able to deliver a third-order rate of convergence for smooth external flows.

4.3. Transonic flow past an ONERA M6 wing

A transonic flow over the ONERA M6 wing at a Mach number of $M_\infty = 0.84$ and an attack of angle of $\alpha = 3.06^\circ$ is considered in this example. This standard test case is chosen to demonstrate that the developed HWENO (P_1P_2) method can effectively

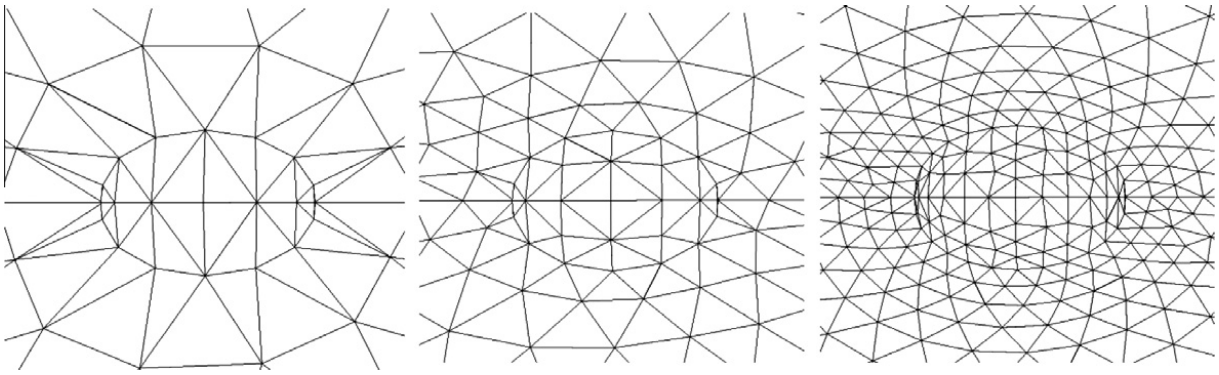

Fig. 3. A series of four successively globally refined tetrahedral meshes for computing a subsonic flow past a sphere at $M_\infty = 0.5$.

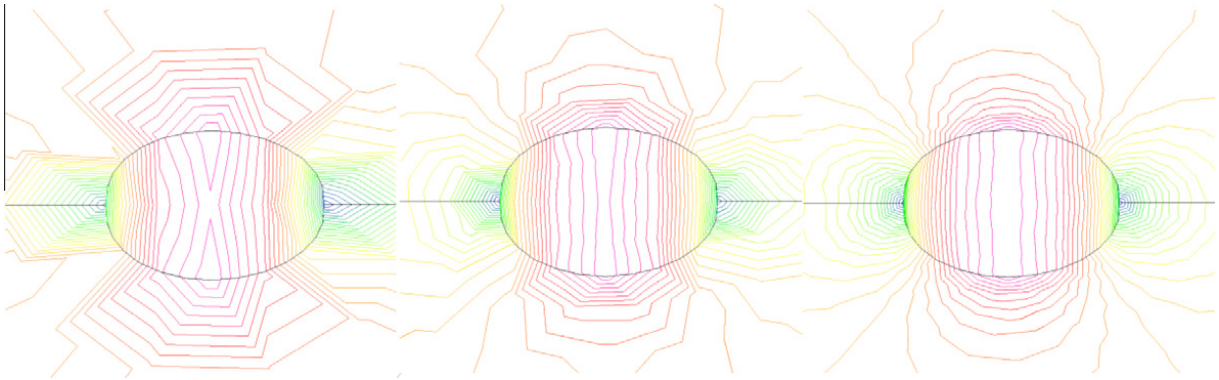

Fig. 4. Computed velocity contours in the flow field obtained by the HWENO (P_1P_2) on a sequence of three successively globally refined unstructured grids for a subsonic flow past a sphere at $M_\infty = 0.5$.

Table 2
L^2-error and order of convergence for the RDG (P_1P_1), WENO (P_1P_2), and HWENO (P_1P_2) methods.

Length scale	RDG (P_1P_1)		WENO (P_1P_2)		HWENO (P_1P_2)	
	L^2-error	Order	L^2-error	Order	L^2-error	Order
7.760E−2	1.783E−2		1.052E−2		1.117E−2	
4.688E−2	5.010E−3	2.519	1.317E−3	4.124	1.503E−3	3.980
2.476E−2	1.232E−3	2.198	1.978E−4	2.964	2.201E−4	3.009

suppress spurious oscillations in the vicinity of strong discontinuities and provide high accuracy solution in comparison with the second-order finite volume method, WENO (P_0P_1) [37]. The flow solutions are presented using the WENO (P_0P_1) method on a fine mesh and the HWENO (P_1P_2) method on a coarse mesh, respectively. The coarse mesh contains 95,266 elements, 18,806 points, and 5,287 boundary points, and the fine one 593,169 elements, 110,282 points, and 19,887 boundary points. Fig. 5 shows the computed pressure contours on the upper wing surface obtained by these two solutions, respectively. The computed pressure coefficients obtained by these two solutions are compared with experimental data [38] at six span-wise stations in Fig. 6. The computed pressure coefficients for the HWENO (P_1P_2) solution are plotted by a straight line connecting the two nodes of each triangle that intersect with the desired cut plane, thus truly reflecting the discontinuous nature of a DG solution. The results obtained by both RDG methods compare closely with the experimental data, except at the root stations, due to the lack of viscous effects. The leading edge suction peak is extremely well captured by both solutions in spite of the coarseness of the grids used in both solutions. However, one can observe that the third-order HWENO (P_1P_2) solution on the coarse mesh is more accurate than the second-order WENO (P_0P_1) solution on a globally refined grid, which is especially evident by judging the entropy production on the surface of the wing at these six span-wise stations as shown in Fig. 7. Note that the entropy production corresponds directly to the error of the numerical methods, as it should be zero everywhere with exception of shock waves where it should increase. The shocks are virtually captured within one cell without any oscillations in the HWENO (P_1P_2) solution, clearly demonstrating the high accuracy and non-oscillatory property of our HWENO (P_1P_2) method.

4.4. Transonic flow past a wing/pylon/finned-store configuration

A transonic flow past a wing/pylon/finned-store configuration reported in Ref. [39] is computed in this test case using HWENO (P_1P_2) method. The configuration consists of a clipped delta wing with a 45° sweep comprised from a constant NACA 64010 symmetric airfoil section. The wing has a root chord of 15 in. a semi-span of 13 in. and a taper ratio of 0.134. The pylon is located at the mid-span station and has a cross-section characterized by a flat plate closed at the leading and trailing edges by a symmetrical ogive shape. The width of the pylon is 0.294 in. The four fins on the store are defined by a constant NACA 0008 airfoil section with a leading-edge sweep of 45° and a truncated tip. The mesh used in the computation contains 319,134 elements, 61,075 grid points, and 14,375 boundary points. The flow solution is presented at a Mach number of 0.95 and an angle of attack of 0°. Fig. 8 shows the computed pressure contours on the upper and lower wing surface, respectively. The computed pressure coefficient distributions are compared with experimental data at two span-wise stations in Fig. 9. The comparison with experimental data is excellent on both upper and lower surface up to 70% chord. As expected from the Euler solution, the computation predicts a shock location that is downstream of that measured by the experiment

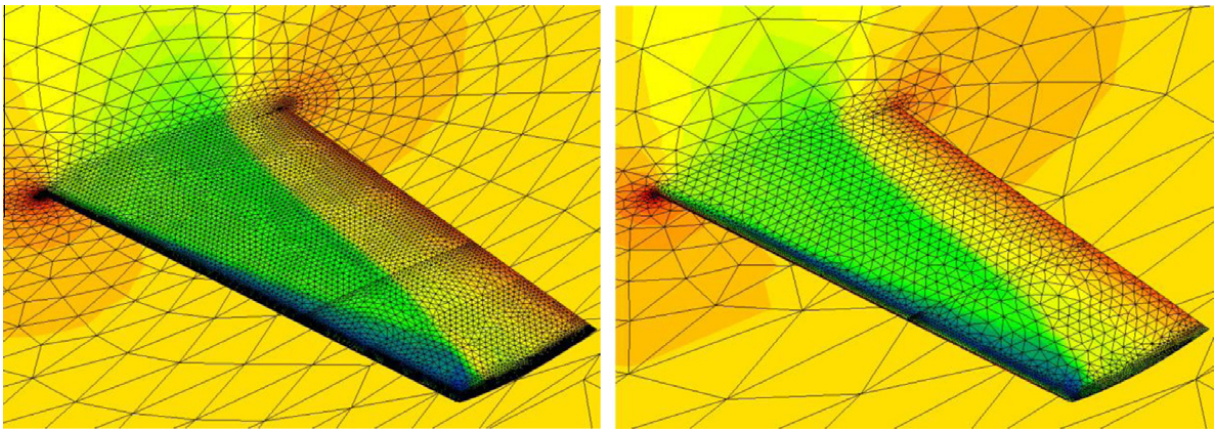

Fig. 5. Computed pressure contours on the unstructured surface mesh obtained by the WENO (P_0P_1) solution on the fine mesh (left, nelem = 593,169, npoin = 110,282, nboun = 19,887), and the HWENO (P_1P_2) solution on the coarse mesh (right, nelem = 95,266, npoin = 18,806, nboun = 5287) for a transonic flow past a M6 wing at $M_\infty = 0.84$, $\alpha = 3.06°$.

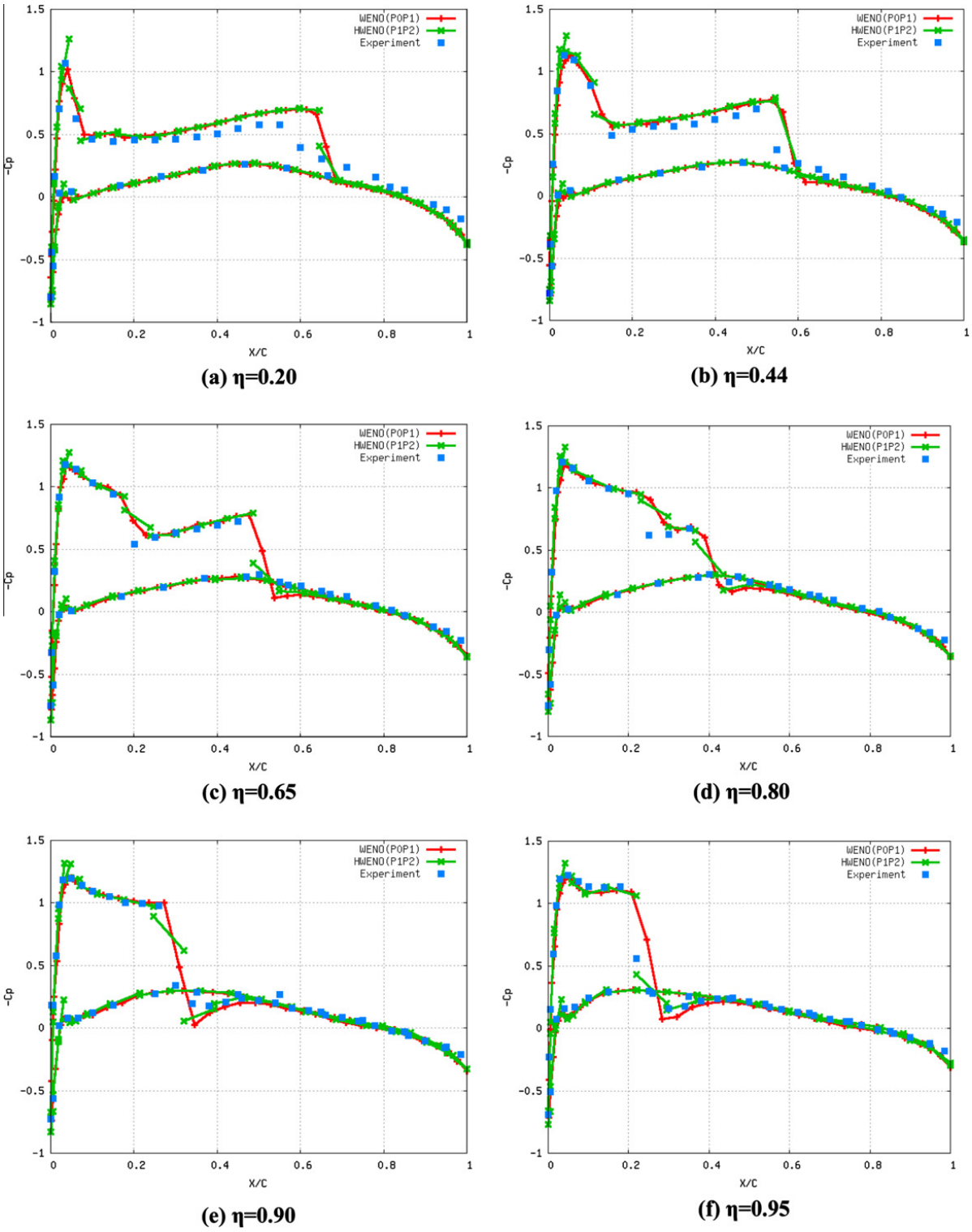

Fig. 6. Comparison of the computed pressure coefficient distributions obtained by WENO (P_0P_1) and HWENO (P_1P_2) solutions with experimental data at six span-wise locations for a transonic flow past the ONERA M6 wing at $M_\infty = 0.84$, and $\alpha = 3.06°$.

due to the lack of viscous effect. Again, our third-order HWENO (P_1P_2) method captures the shock waves very sharply within one cell without any visible under-and over-shoots.

4.5. Transonic flows past a Boeing 747 aircraft

Finally, a transonic flow past a complete Boeing 747 aircraft is presented in this test case. The 747 configuration includes the fuselage, wing, horizontal and vertical tails, under-wing pylons, and flow-through engine nacelle. The mesh used in the

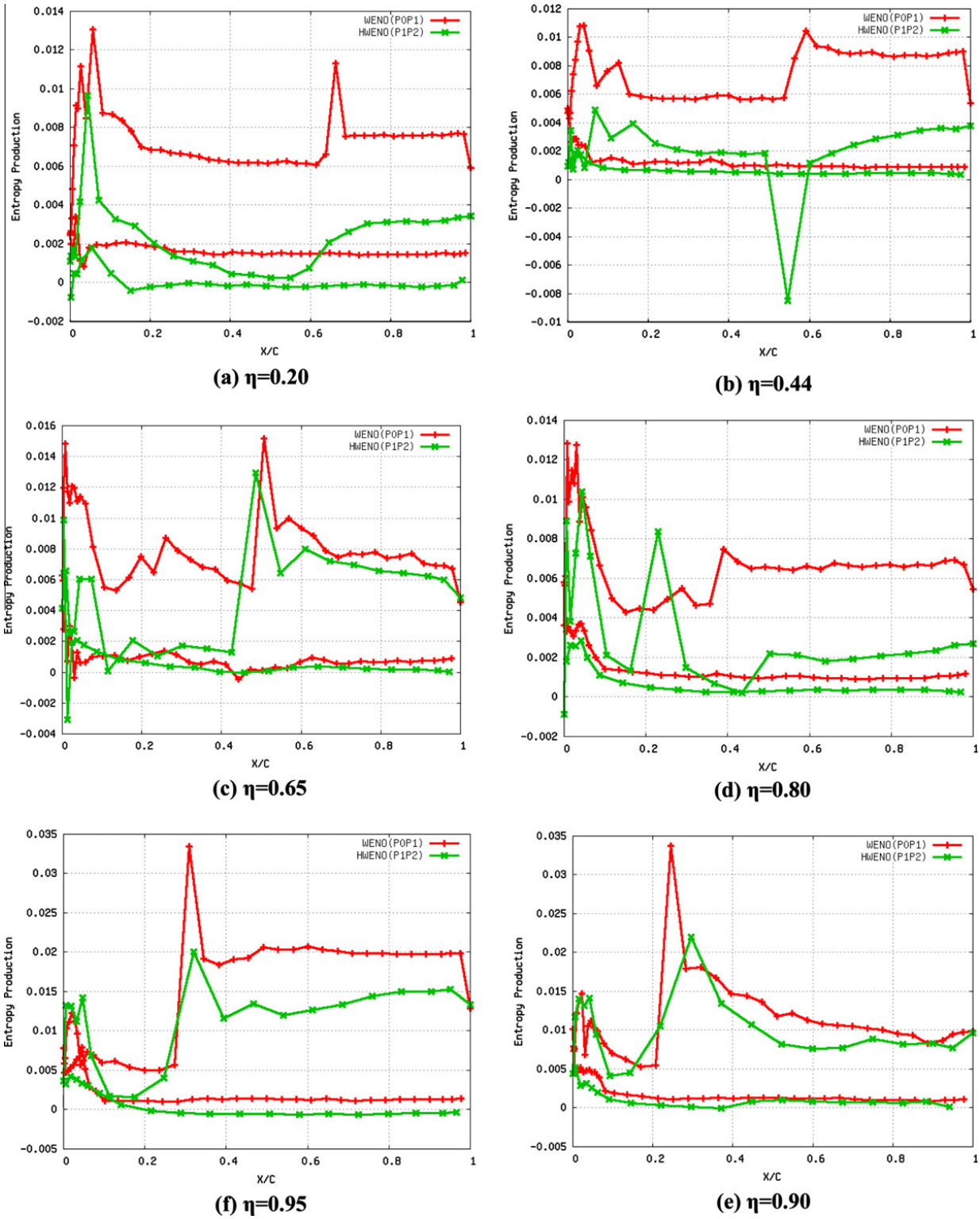

Fig. 7. Comparison of the computed entropy production distributions obtained by the WENO (P_0P_1) and HWENO (P_1P_2) solutions at six span-wise locations for a transonic flow past the ONERA M6 wing at $M_\infty = 0.84$, and $\alpha = 3.06°$.

computation contains 48,851 grid points, 253,577 elements, and 11,802 boundary points for the half-span airplane. The solution is computed at a free stream of Mach number of 0.85 and an angle of attack of 2°. The computed Mach number contours on the surface of the airplane, along with the surface mesh, are shown in Fig. 10. One can see that the shock waves on the upper surface of the wing are captured well within one cell, confirming the accuracy and robustness of the HWENO (P_1P_2) method for computing complicated flows of practical importance.

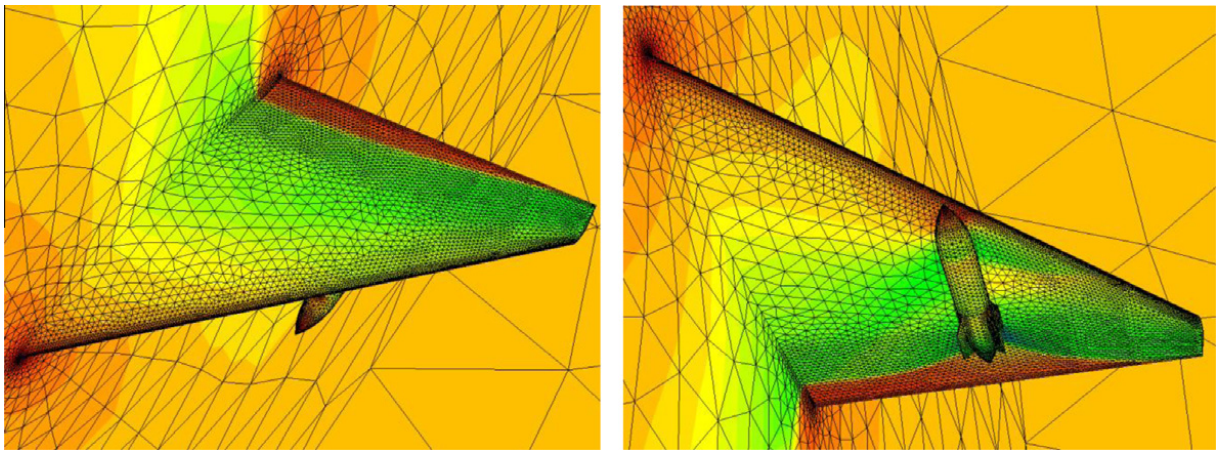

Fig. 8. Computed pressure contours on the unstructured surface mesh obtained by the HWENO (P_1P_2) solution (nelem = 319,134, npoin = 61,075, nboun = 14,373) for a transonic flow past a wing/pylon/finned-store configuration at $M_\infty = 0.95$, and $\alpha = 0°$.

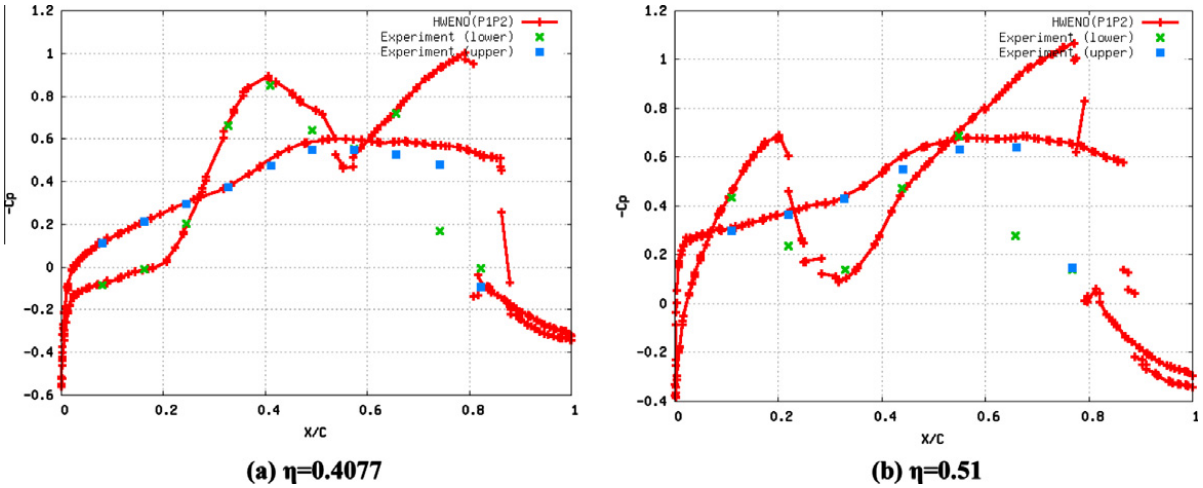

Fig. 9. Comparison of the computed pressure coefficient distributions with experimental data at two span-wise locations for a transonic flow past a wing/pylon/finned-store configuration at $M_\infty = 0.95$, and $\alpha = 0°$.

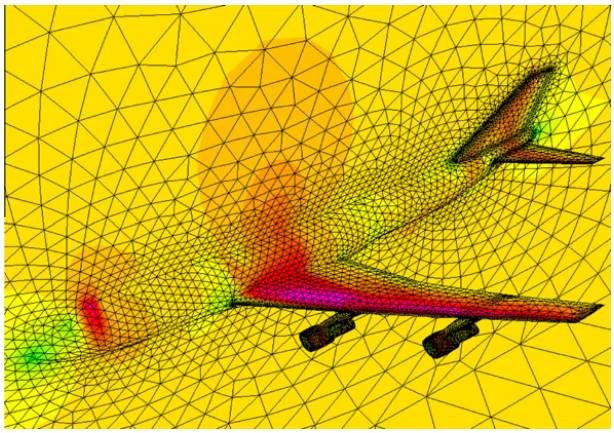

Fig. 10. Computed Mach number contours and unstructured surface mesh for transonic flow past a complete B747 aircraft (nelem = 253,577, npoin = 48,851, nboun = 11,802) at $M_\infty = 0.85$, and $\alpha = 2°$.

5. Conclusions

A reconstructed discontinuous Galerkin method based on a hierarchical Hermite WENO reconstruction, HWENO (P_1P_2), has been presented for solving the compressible Euler equations on tetrahedral grids. The HWENO (P_1P_2) method is designed not only to enhance the accuracy of the discontinuous Galerkin method, but also to avoid non-physical oscillations in the vicinity of discontinuities. A number of numerical experiments for a variety of flow conditions have been conducted to demonstrate the accuracy, robustness, and non-oscillatory performance of the HWENO (P_1P_2) method. The numerical results obtained indicate that the developed HWENO (P_1P_2) method is able to provide sharp resolution of shock waves without over- and under-shoots for flows with strong discontinuities and achieve the designed third-order of accuracy for smooth flows: one order accuracy higher than the underlying DG method, thus significantly increasing the accuracy of the underlying DG method without significant increase in computing costs and memory requirements.

Acknowledgments

This research is partially supported using funding received from the DOE Office of Nuclear Energy's Nuclear Engineering University Program. The first author would like to acknowledge the partial support for this work provided by the fundamental research program of DTRA under Grant No. HDTR1-10-1-0.123. Dr. Suhithi Peiris serves as the technical monitor.

References

[1] W.H. Reed, T.R. Hill, Triangular Mesh Methods for the Neutron Transport Equation, Los Alamos Scientific Laboratory Report, LA-UR-73-479, 1973.
[2] B. Cockburn, S. Hou, C.W. Shu, TVD Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case, Mathematics of Computation, 55 (1990) 545–581 (P.L. George, Automatic Mesh Generation, John Wiley & Sons, 1991).
[3] B. Cockburn, C.W. Shu, The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional system, Journal of Computational Physics 141 (1998) 199–224.
[4] B. Cockburn, G. Karniadakis, C.W. Shu, The development of discontinuous Galerkin method, in: B. Cockburn, G.E. Karniadakis, C.W. Shu (Eds.), Discontinuous Galerkin Methods, Theory, Computation, and Applications,, Lecture Notes in Computational Science and Engineering, Springer-Verlag, New York, 2000, pp. 5–50. vol. 11.
[5] F. Bassi, S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, Journal of Computational Physics 138 (1997) 251–285.
[6] H.L. Atkins, C.W. Shu, Quadrature free implementation of discontinuous Galerkin method for hyperbolic equations, AIAA Journal 36 (5) (1998).
[7] F. Bassi, S. Rebay, GMRES discontinuous Galerkin solution of the compressible Navier–Stokes equations, in: B. Cockburn, G.E. Karniadakis, C.W. Shu (Eds.), Discontinuous Galerkin Methods, Theory, Computation, and Applications, Lecture Notes in Computational Science and Engineering, Springer-Verlag, New York, 2000, pp. 197–208. vol. 11.
[8] T.C. Warburton, G.E. Karniadakis, A discontinuous Galerkin method for the viscous MHD equations, Journal of Computational Physics 152 (1999) 608–641.
[9] J.S. Hesthaven, T. Warburton, Nodal discontinuous Galerkin methods: algorithms, analysis, and applications, Texts in Applied Mathematics 56 (2008).
[10] P. Rasetarinera, M.Y. Hussaini, An efficient implicit discontinuous spectral Galerkin method, Journal of Computational Physics 172 (2001) 718–738.
[11] B.T. Helenbrook, D. Mavriplis, H.L. Atkins, Analysis of *p*-multigrid for continuous and discontinuous finite element discretizations, AIAA Paper 2003-3989, 2003.
[12] K.J. Fidkowski, T.A. Oliver, J. Lu, D.L. Darmofal, *p*-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations, Journal of Computational Physics 207 (1) (2005) 92–113.
[13] H. Luo, J.D. Baum, R. Löhner, A discontinuous Galerkin method using Taylor basis for compressible flows on arbitrary grids, Journal of Computational Physics 227 (20) (2008) 8875–8893.
[14] H. Luo, J.D. Baum, R. Löhner, On the computation of steady-state compressible flows using a discontinuous Galerkin method, International Journal for Numerical Methods in Engineering 73 (5) (2008) 597–623.
[15] H. Luo, J.D. Baum, R. Löhner, A Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids, Journal of Computational Physics 225 (1) (2007) 686–713.
[16] H. Luo, J.D. Baum, R. Löhner, A *p*-multigrid discontinuous Galerkin method for the Euler equations on unstructured grids, Journal of Computational Physics 211 (2) (2006) 767–783.
[17] H. Luo, J.D. Baum, R. Löhner, Fast, *p*-multigrid discontinuous Galerkin method for compressible flows at all speeds, AIAA Journal 46 (3) (2008) 635–652.
[18] M. Dumbser, D.S. Balsara, E.F. Toro, C.D. Munz, A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes, Journal of Computational Physics 227 (2008) 8209–8253.
[19] M. Dumbser, O. Zanotti, Very high order PNPM schemes on unstructured meshes for the resistive relativistic MHD equations, Journal of Computational Physics 228 (2009) 6991–7006.
[20] M. Dumbser, Arbitrary high order PNPM schemes on unstructured meshes for the compressible Navier–Stokes equations, Computers and Fluids 39 (2010) 60–76.
[21] F. Bassi, S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations, Journal of Computational Physics 131 (1997) 267–279.
[22] F. Bassi, S. Rebay, Discontinuous Galerkin solution of the Reynolds-averaged Navier–Stokes and k-ω turbulence model equations, Journal of Computational Physics 34 (2005) 507–540.
[23] B. Cockburn, C.W. Shu, The local discontinuous Galerkin method for time-dependent convection-diffusion system, SIAM, Journal of Numerical Analysis 16 (2001) 301–318.
[24] C.E. Baumann, J.T. Oden, A discontinuous *hp* finite element method for the Euler and Navier–Stokes equations, International Journal for Numerical Methods in Fluids 31 (1999).
[25] H. Luo, L. Luo, K. Xu, A discontinuous Galerkin method based on a BGK scheme for the Navier–Stokes equations on arbitrary grids, Advances in Applied Mathematics and Mechanics 1 (3) (2009) 301–318.
[26] H. Luo, L. Luo, R. Nourgaliev, A reconstructed discontinuous Galerkin method for the Euler equations on arbitrary grids, Communication in Computational Physics 12 (5) (2012) 1495–1519.
[27] H. Luo, L. Luo, R. Nourgaliev, V.A. Mousseau, N. Dinh, A reconstructed discontinuous Galerkin method for the compressible Navier–Stokes equations on arbitrary grids, Journal of Computational Physics 229 (2010) 6961–6978.
[28] H. Luo, L. Luo, A. Ali, R. Nourgaliev, C. Cai, A parallel, reconstructed discontinuous Galerkin method for the compressible flows on arbitrary grids, Communication in Computational Physics 9 (2) (2011) 363–389.

[29] L.P. Zhang, W. Liu, L.X. He, X.G. Deng, H.X. Zhang, A class of hybrid DG/FV methods for conservation laws I: basic formulation and one-dimensional system, Journal of Computational Physics 23 (4) (2012) 1081–1103.
[30] L.P. Zhang, W. Liu, L.X. He, X.G. Deng, H.X. Zhang, A class of hybrid DG/FV methods for conservation laws II: two dimensional cases, Journal of Computational Physics 23 (4) (2012) 1104–1120.
[31] H. Luo, Y. Xia, R. Nourgaliev, C. Cai, A class of reconstructed discontinuous Galerkin methods for the compressible flows on arbitrary grids, AIAA-2011-0199, 2011.
[32] H. Luo, H. Xiao, R. Nourgaliev, C. Cai, A comparative study of different reconstruction schemes for reconstructed discontinuous Galerkin methods for the compressible flows on arbitrary grids, AIAA-2011-3839, 2011.
[33] D.F. Haider, J.P. Croisille, B. Courbet, Stability analysis of the cell centered finite-volume MUSCL method on unstructured grids, Numirische Mathematik 113 (4) (2009) 555–600.
[34] D. Balsara, C. Altmann, C.D. Munz, M. Dumbser, A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG + HWENO schemes, Journal of Computational Physics 226 (2007) 586–620.
[35] Z. Xu, Y. Liu, H. Du, C.W. Shu, Point-wise hierarchical reconstruction for discontinuous Galerkin and finite volume method for solving conservation laws, Journal of Computational Physics 230 (2011) 6843–6865.
[36] H. Luo, Y. Xia, S. Li, R. Nourgaliev, C. Cai, A Hermite WENO reconstruction-based discontinuous Galerkin method for the Euler equations on tetrahedral grids, Journal of Computational Physics 231 (2012) 5489–5503.
[37] S. Spiegel, H. Luo, A finite volume method based on WENO reconstruction for compressible flows on hybrid grids, submitted for publication at 51th AIAA Sciences Meeting, 2013.
[38] V. Schmitt, E. Charpin, Pressure distributions on the ONERA M6-wing at transonic mach numbers, experimental data base for computer program assessment, AGARD AR-138, 1979.
[39] E.R. Ilieim, CFD wing/pylon/finned store mutual interference wind tunnel experiment, AEDC-TSR-91-P4, Arnold Engineering Development Center, Arnold AFB, TN, January 1991.
[40] M. Dumbser, M. Kaser, Arbitrary high order non-oscillatory finite volume schemes on unstructured meshes for linear hyperbolic system, Journal of Computational Physics 221 (2007) 693–723.
[41] R. Löhner, P. Parikh, Three-dimensional grid generation by the advancing front method, International Journal for Numerical Methods in Fluids (8) (1988) 1135–1149.
[42] D. Sharov, H. Luo, J.D. Baum, R. Löhner, Unstructured Navier–Stokes grid generation at corners and ridges, International Journal for Numerical Methods in Fluids 43 (2003) 717–728.
[43] H. Luo, J.D. Baum, R. Löhner, Extension of HLLC scheme for flows at all speeds, AIAA Journal 43 (6) (2005) 1160–1166.

Appendix 2

Representative publications regarding the development of implicit methods for the rDG methods

Computers & Fluids 98 (2014) 134–151

Contents lists available at ScienceDirect

Computers & Fluids

journal homepage: www.elsevier.com/locate/compfluid

A set of parallel, implicit methods for a reconstructed discontinuous Galerkin method for compressible flows on 3D hybrid grids

Yidong Xia [a,*], Hong Luo [a], Megan Frisbey [a], Robert Nourgaliev [b]

[a] Department of Mechanical and Aerospace Engineering, North Carolina State University, Raleigh, NC 27695, USA
[b] B Division, Weapons and Complex Integration, Lawrence Livermore National Laboratory, Livermore, CA 94550, USA

ARTICLE INFO

Article history:
Received 27 July 2013
Received in revised form 17 January 2014
Accepted 17 January 2014
Available online 1 February 2014

Keywords:
Discontinuous Galerkin
Hierarchical WENO reconstruction
Implicit methods
Automatic differentiation
Parallel computing

ABSTRACT

A set of implicit methods are proposed for a third-order hierarchical WENO reconstructed discontinuous Galerkin method for compressible flows on 3D hybrid grids. An attractive feature in these methods are the application of the Jacobian matrix based on the P_1 element approximation, resulting in a huge reduction of memory requirement compared with DG (P_2). Also, three approaches — analytical derivation, divided differencing, and automatic differentiation (AD) are presented to construct the Jacobian matrix respectively, where the AD approach shows the best robustness. A variety of compressible flow problems are computed to demonstrate the fast convergence property of the implemented flow solver. Furthermore, an SPMD (single program, multiple data) programming paradigm based on MPI is proposed to achieve parallelism. The numerical results on complex geometries indicate that this low-storage implicit method can provide a viable and attractive DG solution for complicated flows of practical importance.

1. Introduction

The discontinuous Galerkin (DG) methods [1–26] have recently become popular for the solution of systems of conservation laws. The DG methods were first introduced for the solution of neutron transport equations [27], and nowadays they are widely used in computational fluid dynamics (CFD), computational acoustics, and computational magneto-hydrodynamics (MHD). A number of attractive features of the DG methods are listed in Ref. [28]. However, the DG methods also have their own weaknesses. Indeed, compared to the finite element (FE) and finite volume (FV) methods, the DG methods require solutions of systems of equations with more unknowns for the same grids. Consequently, DG are recognized as expensive in terms of both computational costs and storage requirements.

In recent years, a number of so-called reconstruction DG schemes are proposed in order to reduce the high costs associated with the standard DG methods, like the recovery-based RDG (P_nP_m) schemes ($n \leqslant m$) by Dumbser et al. [29–31], the hybrid HWENO + DG schemes by Balsara et al. [32], the least-squares reconstruction DG schemes by Luo et al. [33–37,28], and the class of Green-Gauss reconstruction hybrid DG/FV schemes by Zhang et al. [38,39]. All of these schemes are able to improve the spatial accuracy of the underlying DG methods without significant extra cost in storage and computing time. On the other side, in order to achieve fast convergence for DG methods, implicit time integration is required. Unfortunately, many high-order implicit DG methods [40,8,41,5,42,9,7,43,44] require a considerable amount of memory to store the Jacobian matrix. Indeed, it is our belief that a lack of efficient solvers is one of the reasons that the application of the DG methods for engineering-type problems does not exist.

Recently the present authors proposed an implicit WENO reconstruction-based DG method for compressible flows on tetrahedral grids [45,46]. In this method, computation of the Jacobian matrix is based on the underlying DG (P_1) method only. The most attractive feature of this implicit method is its low-storage requirement due to its DG (P_1)-like linear system of equations, which requires less than 1/6 of the memory of the implicit DG (P_2) method to achieve the same order of accuracy. Furthermore, in order to alleviate the excessive human labor for constructing a well approximated Jacobian matrix, the present authors introduced an implicit method based on automatic differentiation for the WENO reconstruction-based DG method on tetrahedral grids [47]. The resulting implicit method is highly robust and efficient according to a variety of test cases. In the present work, this method is extended to 3D hybrid grids. Besides, the performance of three approaches for constructing the Jacobian matrix: (1) analytical

* Corresponding author. Tel.: +1 919 534 5645.
E-mail address: yxia2@ncsu.edu (Y. Xia).

http://dx.doi.org/10.1016/j.compfluid.2014.01.023
0045-7930/

derivation, (2) divided differencing, and (3) automatic differentiation is analyzed, respectively. A parallel strategy is proposed and implemented based on a message passing interface (MPI) programming paradigm. A variety of compressible flow problems for a wide range of flow conditions in 3D configurations are computed to demonstrate the performance of the developed implicit methods.

The remainder of this paper is organized as follows. The governing equations are described in Section 2. The discontinuous Galerkin discretization and the hierarchical WENO reconstruction schemes are briefly introduced in Sections 3 and 3.2. The implicit time integration method is given in Section 4. A set of methods to construct the Jacobian matrix are presented and discussed in Section 5. The parallelization strategy is illustrated in Section 6. The numerical experiments are reported in Section 7. Finally, concluding remarks are given in Section 8.

2. Governing equations

The Navier–Stokes equations governing unsteady compressible viscous flows can be expressed as

$$\frac{\partial \mathbf{U}(x,t)}{\partial t}+\frac{\partial \mathbf{F}_k(\mathbf{U}(x,t))}{\partial x_k}=\frac{\partial \mathbf{G}_k(\mathbf{U}(x,t),\nabla\mathbf{U}(x,t))}{\partial x_k} \tag{1}$$

where the summation convention has been used. The conservative variable vector $\mathbf{U}$, advective flux vector $\mathbf{F}$, and viscous flux vector $\mathbf{G}$ are defined by

$$\mathbf{U}=\begin{pmatrix}\rho\\ \rho u_i\\ \rho e\end{pmatrix}\quad \mathbf{F}_j=\begin{pmatrix}\rho u_j\\ \rho u_i u_j+p\delta_{ij}\\ u_j(\rho e+p)\end{pmatrix}\quad \mathbf{G}_j=\begin{pmatrix}0\\ \tau_{ij}\\ u_i\tau_{ij}+q_j\end{pmatrix} \tag{2}$$

Here ρ, p, and e denote the density, pressure, and specific total energy of the fluid, respectively, and u_i is the velocity of the flow in the coordinate direction x_i. The pressure can be computed from the equation of state

$$p=(\gamma-1)\rho\left(e-\frac{1}{2}(u^2+v^2+w^2)\right) \tag{3}$$

which is valid for perfect gas. The ratio of the specific heats γ is assumed to be constant and equal to 1.4. The viscous stress tensor τ_{ij} and heat flux vector q_j are given by

$$\tau_{ij}=\mu\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)-\frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\delta_{ij}\quad q_j=\frac{1}{\gamma-1}\frac{\mu}{Pr}\frac{\partial T}{\partial x_j} \tag{4}$$

In the above equations, T is the temperature of the fluid, Pr the laminar Prandtl number, which is taken as 0.7 for air. μ represents the molecular viscosity, which can be determined through Sutherlands law

$$\frac{\mu}{\mu_0}=\left(\frac{T}{T_0}\right)^{\frac{3}{2}}\frac{T_0+S}{T+S} \tag{5}$$

μ_0 denotes the viscosity at the reference temperature T_0 and $S=110$ K. The temperature of the fluid T is determined by

$$T=\gamma\frac{P}{\rho} \tag{6}$$

Neglecting viscous effects, the left-hand-side of Eq. (1) represents the Euler equations governing unsteady compressible inviscid flows.

3. Reconstructed discontinuous Galerkin method

3.1. Discontinuous Galerkin spatial discretization

Eq. (1) can be discretized using a discontinuous Galerkin finite element formulation. First, we assume that the domain Ω is subdivided into a collection of non-overlapping arbitrary elements Ω_e in 3D, and introduce the following broken Sobolev space V_h^p

$$V_h^p=\left\{v_h\in\left[L^2(\Omega)\right]^m:\ v_h|_{\Omega_e}\in\left[V_p^m\right]\quad\forall\Omega_e\in\Omega\right\} \tag{7}$$

which consists of discontinuous vector polynomial functions of degree p, and where m is the dimension of the unknown vector and V_p is the space of all polynomials of degree $\leqslant p$. To formulate the discontinuous Galerkin method, we introduce the following weak formulation, which is obtained by multiplying Eq. (1) by a test function W, integrating over an element Ω_e, and then performing an integration by parts: find $\mathbf{U}\in V_p$ such as

$$\begin{aligned}&\frac{d}{dt}\int_{\Omega_e}\mathbf{U}W\,d\Omega+\int_{\Gamma_e}\mathbf{F}_k(\mathbf{U})\cdot\mathbf{n}_k W\,d\Gamma-\int_{\Omega_e}\mathbf{F}_k(\mathbf{U})\cdot\frac{\partial W}{\partial x_k}\,d\Omega\\ &=\int_{\Gamma_e}\mathbf{G}_k(\mathbf{U},\nabla\mathbf{U})\cdot\mathbf{n}_k W\,d\Gamma-\int_{\Omega_e}\mathbf{G}_k(\mathbf{U},\nabla\mathbf{U})\cdot\frac{\partial W}{\partial x_k}\,d\Omega\quad\forall\mathbf{W}\in V^p\end{aligned} \tag{8}$$

where $\mathbf{U}$ and W are represented by piecewise polynomial functions of degrees p, which are discontinuous between the cell interfaces, and $\mathbf{n}_k$ the unit outward normal vector to the Γ_e: the boundary of Ω_e. Assume that B_i is the basis of polynomial function of degrees p, this is then equivalent to the following system of N equations,

$$\begin{aligned}&\frac{d}{dt}\int_{\Omega_e}\mathbf{U}B_i\,d\Omega+\int_{\Gamma_e}\mathbf{F}_k(\mathbf{U})\cdot\mathbf{n}_k B_i\,d\Gamma-\int_{\Omega_e}\mathbf{F}_k(\mathbf{U})\cdot\frac{\partial B_i}{\partial x_k}\,d\Omega\\ &=\int_{\Gamma_e}\mathbf{G}_k(\mathbf{U},\nabla\mathbf{U})\cdot\mathbf{n}_k B_i\,d\Gamma-\int_{\Omega_e}\mathbf{G}_k(\mathbf{U},\nabla\mathbf{U})\cdot\frac{\partial B_i}{\partial x_k}\,d\Omega\quad 1\leqslant i\leqslant N\end{aligned} \tag{9}$$

where N is the dimension of the polynomial space. Since the numerical solution $\mathbf{U}$ is discontinuous between element interfaces, the interface fluxes are not uniquely defined. The flux function $\mathbf{F}_k(\mathbf{U})\cdot\mathbf{n}_k$ appearing in the second terms of Eq. (9) is replaced by a numerical Riemann flux function $\mathbf{H}_k(\mathbf{U}_h^L,\mathbf{U}_h^R,\mathbf{n}_k)$ where $\mathbf{U}_h^L$ and $\mathbf{U}_h^R$ are the conservative state vectors at the left and right side of the element boundary. This scheme is called discontinuous Galerkin method of degree p, or in short notation DG (p) method. By simply increasing the degree p of the polynomials, the DG methods of corresponding higher order are obtained. In the present work, the inviscid flux is evaluated by the HLLC [48] scheme and the viscous flux by the Bassi–Rebay II scheme [5], respectively. For nodal discontinuous Galerkin methods, numerical polynomial solutions $\mathbf{U}_h$ in each element are expressed using either standard Lagrange finite element or hierarchical node-based basis like

$$\mathbf{U}_h=\sum_{i=1}^{N}\mathbf{U}_i B_i(x) \tag{10}$$

where B_i is the finite element basis function. The resulting unknowns to be solved are the variables at the nodes $\mathbf{U}_i$. In the DG method of our work, the numerical polynomial solutions are represented using a Taylor series expansion at the cell centroid and normalized in order to improve the conditioning of the system matrix Eq. (9). For example, the linear polynomial P_1 solutions of the underlying DG (P_1) method used in the present work, consist of cell-averaged values $\tilde{\mathbf{U}}$ and their normalized first derivatives $\mathbf{U}_x=\frac{\partial\mathbf{U}}{\partial x}|_c\Delta x$, $\mathbf{U}_y=\frac{\partial\mathbf{U}}{\partial y}|_c\Delta y$, $\mathbf{U}_z=\frac{\partial\mathbf{U}}{\partial z}|_c\Delta z$ at the center of the cell, as expressed below,

$$\mathbf{U}_h^{P_1}=\tilde{\mathbf{U}}B_1+\mathbf{U}_x B_2+\mathbf{U}_y B_3+\mathbf{U}_z B_4 \tag{11}$$

where the four basis functions are as below.

$$B_1 = 1 \quad B_2 = \frac{x - x_c}{\Delta x} \quad B_3 = \frac{y - y_c}{\Delta y} \quad B_4 = \frac{z - z_c}{\Delta z} \tag{12}$$

where $\Delta x = 0.5(x_{max} - x_{min})$, $\Delta y = 0.5(y_{max} - y_{min})$, $\Delta z = 0.5(z_{max} - z_{min})$. $x_{max}, y_{max}, z_{max}$ and $x_{min}, y_{min}, z_{min}$ are the maximum and minimum vertex coordinates of the cell Ω_e, respectively. The above normalization is especially important to make the system matrix less stiff for higher-order discontinuous Galerkin approximations.

This formulation has a number of attractive, distinct, and useful features. First, cell-averaged variables and their derivatives are handily available in this formulation. This makes the implementation of both in-cell and inter-cell reconstruction schemes straightforward and simple [33,35,49,39,50]. Secondly, the Taylor basis is hierarchic, which greatly facilitates the implementation of *p*-multigrid methods [51,52] and *p*-refinement. Thirdly, the same basis functions are used for any shapes of elements: tetrahedron, pyramid, prism, and hexahedron. This makes the implementation of DG methods on arbitrary grids straightforward.

3.2. Hierarchical WENO reconstruction scheme

A third-order hierarchical WENO reconstruction scheme based on the reconstructed discontinuous Galerkin method [34,50] designed by the authors earlier for tetrahedral grids [28] is extended to 3D hybrid grids in this work. This scheme adopts a hierarchical reconstruction strategy [53] (successively from high order to low order), where the second and first derivatives are reconstructed in a hierarchical manner.

Firstly, the six normalized second derivatives $\mathbf{U}^{\mathrm{R}}_{xx} = \frac{\partial^2 \mathbf{U}}{\partial x^2}|_c \Delta x^2$, $\mathbf{U}^{\mathrm{R}}_{yy} = \frac{\partial^2 \mathbf{U}}{\partial y^2}|_c \Delta y^2$, $\mathbf{U}^{\mathrm{R}}_{zz} = \frac{\partial^2 \mathbf{U}}{\partial z^2}|_c \Delta z^2$, $\mathbf{U}^{\mathrm{R}}_{xy} = \frac{\partial^2 \mathbf{U}}{\partial x \partial y}|_c \Delta x \Delta y$, $\mathbf{U}^{\mathrm{R}}_{xz} = \frac{\partial^2 \mathbf{U}}{\partial x \partial z}|_c \Delta x \Delta z$, $\mathbf{U}^{\mathrm{R}}_{yz} = \frac{\partial^2 \mathbf{U}}{\partial y \partial z}|_c \Delta y \Delta z$ are reconstructed using a least-squares method [36] from the underlying linear polynomial (P_1) discontinuous Galerkin solution, and the reconstructed quadratic polynomial (P_2) solution is expressed as follows:

$$\mathbf{U}_h^{\mathrm{RP}_2} = \mathbf{U}_h^{\mathrm{P}_1} + \mathbf{U}^{\mathrm{R}}_{xx} B_5 + \mathbf{U}^{\mathrm{R}}_{yy} B_6 + \mathbf{U}^{\mathrm{R}}_{zz} B_7 + \mathbf{U}^{\mathrm{R}}_{xy} B_8 + \mathbf{U}^{\mathrm{R}}_{xz} B_9 + \mathbf{U}^{\mathrm{R}}_{yz} B_{10} \tag{13}$$

where the six additional basis functions are

$$\begin{aligned} B_5 &= \frac{B_2^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{B_2^2}{2}\, d\Omega \quad B_6 = \frac{B_3^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{B_3^2}{2}\, d\Omega \\ B_7 &= \frac{B_4^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{B_4^2}{2}\, d\Omega \quad B_8 = B_2 B_3 - \frac{1}{\Omega_e}\int_{\Omega_e} B_2 B_3\, d\Omega \\ B_9 &= B_2 B_4 - \frac{1}{\Omega_e}\int_{\Omega_e} B_2 B_4\, d\Omega \quad B_{10} = B_3 B_4 - \frac{1}{\Omega_e}\int_{\Omega_e} B_3 B_4\, d\Omega \end{aligned} \tag{14}$$

The final second derivatives are then obtained using a WENO reconstruction as expressed below, which is necessary to ensure the linear stability of the RDG method on unstructured elements,

$$\left.\frac{\partial^2 \mathbf{U}}{\partial x_m \partial x_n}\right|_i^{\mathrm{WENO}} = \sum_{k=1}^{1+N_{es}} w_k^{(2)} \left.\frac{\partial^2 \mathbf{U}}{\partial x_m \partial x_n}\right|_k^{\mathrm{R}} \tag{15}$$

where N_{es} denotes the number of face-neighboring cells (equals to 4, 5, 5, and 6 for a tetrahedron, pyramid, prism and hexahedron, respectively). The procedure of computing the normalized nonlinear weights $w_k^{(2)}$ is described in Ref. [36]. The resulting RDG ($\mathrm{P}_1\mathrm{P}_2$) method is referred to as WENO ($\mathrm{P}_1\mathrm{P}_2$).

Secondly, the first derivatives are reconstructed using a WENO reconstruction in order to eliminate non-physical oscillations in the vicinity of strong discontinuities and thus maintain the nonlinear stability,

$$\left.\frac{\partial \mathbf{U}}{\partial x_m}\right|_i^{\mathrm{WENO}} = \sum_{k=1}^{N_{es}} w_k^{(1)} \left.\frac{\partial \mathbf{U}}{\partial x_m}\right|_k \tag{16}$$

where the procedure of computing the normalized nonlinear weights $w_k^{(1)}$ are described in Ref. [28]. The resulting RDG ($\mathrm{P}_1\mathrm{P}_2$) method based on this hierarchical WENO reconstruction is termed as HWENO ($\mathrm{P}_1\mathrm{P}_2$). The present choice of reconstruction stencils is symmetric, and compact, as only von Neumann neighbors are involved, as demonstrated in Fig. 1. This means that no additional data structure is required for our HWENO ($\mathrm{P}_1\mathrm{P}_2$) scheme.

In the HWENO ($\mathrm{P}_1\mathrm{P}_2$) RDG method, the reconstructed quadratic polynomial solution is then used to compute the domain and boundary integrals of the underlying DG (P_1) method in Eq. (9). As demonstrated in Ref. [28], this resulting HWENO ($\mathrm{P}_1\mathrm{P}_2$) RDG method is also able to achieve the designed third-order of accuracy for smooth flows at a moderate increase of computing costs in comparison with the underlying DG (P_1) method. The extra costs are mainly due to the reconstruction, which is relatively cheap in comparison to the evaluation of fluxes, and an extra Gauss quadrature point, which is required to calculate the domain integrals for the tetrahedral cell (5 quadrature points), as demonstrated in Table 1. In comparison to DG (P_2), this represents a significant saving in terms of flux evaluations. Most importantly, the number of degrees of freedom is significantly reduced, which leads to a significant reduction in memory requirements, and from which implicit methods will benefit tremendously. As a result, the implicit methods for HWENO ($\mathrm{P}_1\mathrm{P}_2$) requires a much lower storage than DG (P_2): 400 versus 2500 words per elemental implicit diagonal matrix (IDM), while both can achieve the same order of accuracy!

4. Implicit temporal discretization

The spatial discretization of the governing equations leads to a system of ordinary differential equations (ODEs) in time and Eq. (9) can be written in an elemental semi-discrete form as

$$\boldsymbol{M}\frac{d\mathbf{U}}{dt} = \mathbf{R}(\mathbf{U}) \tag{17}$$

where $\mathbf{U} = (\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_k, \ldots, \mathbf{U}_{\mathrm{Nelem}})^T$ is the global solution vector of Ndegr × Netot × Nelem DOFs to be evolved in time. By applying the backward Euler scheme to Eq. (17), one obtains

$$\boldsymbol{M}\frac{(\mathbf{U}^{n+1} - \mathbf{U}^n)}{\Delta t} = \mathbf{R}(\mathbf{U}^{n+1}) \tag{18}$$

which is a system of nonlinear equations for $\mathbf{U}^{n+1}$. In order to solve this type of equations, we can linearize $\mathbf{R}$ with respect to $\mathbf{U}$ at the current time-step

$$\mathbf{R}(\mathbf{U}^{n+1}) \approx \mathbf{R}(\mathbf{U}^n) + \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^n (\mathbf{U}^{n+1} - \mathbf{U}^n) \tag{19}$$

where $(\frac{\partial \mathbf{R}}{\partial \mathbf{U}})^n$ is the Jacobian matrix of the system, and denoted symbolically as $\mathbf{J}(\mathbf{U}^n)$. If we replace the right-hand-side term in Eq. (18) with Eq. (19), we can get a delta form of the linear system of equations as follows:

$$\mathbf{A}\Delta\mathbf{U}^n = \left(\frac{M}{\Delta t} - \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^n\right)\Delta\mathbf{U}^n = \mathbf{R}(\mathbf{U}^n) \tag{20}$$

where $\mathbf{A}$ is the left-hand-side matrix, and Δt is the time increment and $\Delta\mathbf{U}^n = \mathbf{U}^{n+1} - \mathbf{U}^n$ is the solution difference between time level n and $n+1$. In the present flow solver, two algorithms are used for the solution of the resulting linear system of equations. One is the SGS (k) (Symmetric Gauss–Seidel) method [54], where k is a pre-set number of sub-iterations. The other one is the LU-SGS (lower–upper SGS) preconditioned GMRES (generalized minimal residual) method [45–47,55], termed as GMRES + LU-SGS. These two methods are well accepted and the implementation details can be found in referred articles above. Also note that the GMRES [56] method is among the most popular and efficient iterative

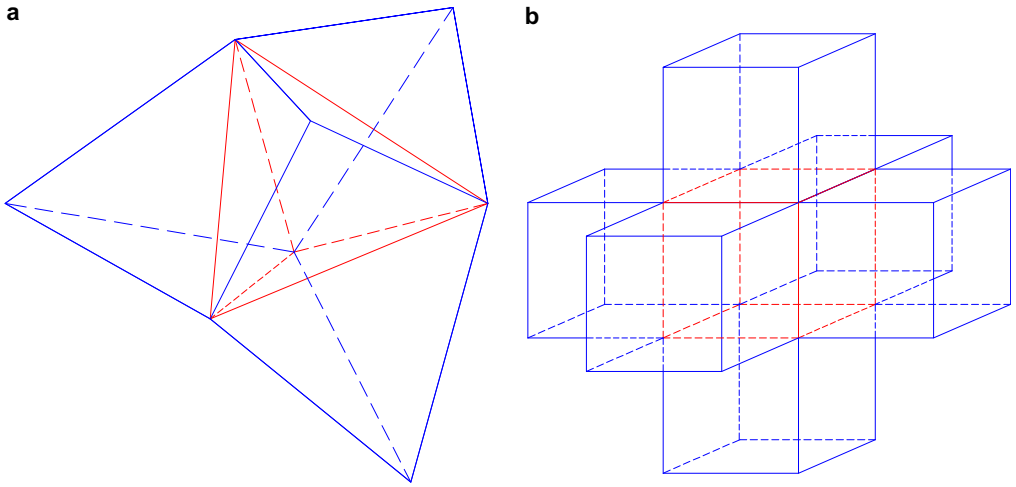

Fig. 1. (a) The tetrahedral cell and its 4 face-neighboring cells. (b) The hexahedral cell and its 6 face-neighboring cells.

Table 1
Cost for DG (P_1), HWENO (P_1P_2) and DG (P_2) on a tetrahedral cell.

	DG (P_1)	HWENO (P_1P_2)	DG (P_2)
N_{QP} for $\int_\Gamma f d\Gamma$	3	4	7
N_{QP} for $\int_\Omega f d\Omega$	4	5	7
Reconstruction	No	Yes	No
Accuracy	$\mathcal{O}(h^2)$	$\mathcal{O}(h^3)$	$\mathcal{O}(h^3)$
Unit mem. for IDM	400	400	2500

algorithms and employed by various authors [57,40,41,58,42,13,59,52,54,5,60,61,8,44].

5. Construction of the Jacobian matrix

In general, four methods are available to construct the Jacobian matrix of the linear system of equations: (1) based on analytical derivation; (2) based on numerical differentiation; (3) based on automatic differentiation; and (4) based on symbolic mathematical software (this is not implemented due to the time-consuming manual operation). The first three methods are to be discussed in detail, respectively.

5.1. Based on analytical derivation

The method based on analytical derivation requires the manual implementation of analytic derivative formulae, which can typically result in very efficient code. However, this method is always tedious, error-prone, and requires great manpower, if no simplification of the system is made. In the case of the HWENO (P_1P_2) RDG method, the exact Jacobian matrix is very hard to obtain, since the HWENO (P_1P_2) reconstruction is highly nonlinear in nature, and thus the linearization of such process might not be easily conducted in an explicit layout. Even in the standard DG method, exact linearization of the viscous flux scheme, e.g., the BR2 scheme, is not trivial work. Thus more than usual, approximate Jacobian matrix is used instead of the exact one in implicit DG methods [48,41,58,42,52,62,13,63,44]. In the present work, the Jacobian matrix for the HWENO (P_1P_2) RDG method can be derived based on the approximate linearization of the underlying DG (P_1) method, that is, $J_{\mathrm{HWENO}(P_1P_2)} \approx J_{\mathrm{DG}(P_1)}$. More specifically, the following two approximations are adopted in face integrals and domain integrals, respectively,

$$\frac{\partial \mathbf{H}_{ij}\left(\mathbf{U}_i^{\mathrm{R}}, \mathbf{U}_j^{\mathrm{R}}, \mathbf{n}_{ij}\right)}{\partial \mathbf{U}_i} = \frac{\partial \mathbf{H}_{ij}\left(\mathbf{U}_i^{\mathrm{R}}, \mathbf{U}_j^{\mathrm{R}}, \mathbf{n}_{ij}\right)}{\partial \mathbf{U}_i^{\mathrm{R}}} \frac{\partial \mathbf{U}_i^{\mathrm{R}}}{\partial \mathbf{U}_i} \approx \frac{\partial \mathbf{H}_{ij}(\mathbf{U}_i, \mathbf{U}_j, \mathbf{n}_{ij})}{\partial \mathbf{U}_i} \tag{21}$$

$$\frac{\partial \mathbf{F}_i\left(\mathbf{U}_i^{\mathrm{R}}\right)}{\partial \mathbf{U}_i} = \frac{\partial \mathbf{F}_i\left(\mathbf{U}_i^{\mathrm{R}}\right)}{\partial \mathbf{U}_i^{\mathrm{R}}} \frac{\partial \mathbf{U}_i^{\mathrm{R}}}{\partial \mathbf{U}_i} \approx \frac{\partial \mathbf{F}_i(\mathbf{U}_i)}{\partial \mathbf{U}_i} \tag{22}$$

where $\mathbf{U}_i^{\mathrm{R}}$ denotes the HWENO (P_1P_2) reconstructed solution. Furthermore, for the inviscid part, the HLLC flux scheme is linearized approximately by assuming the wavespeed as constant in face integrals, which is called the frozen wavespeed version of implicit HLLC scheme in Ref. [48]. For the viscous part, the BR2 scheme is linearized by treating the local lift operator in face integrals, the global lift operator in domain integrals, and the molecular viscosity as constant, respectively.

5.2. Based on numerical differentiation

The method based on numerical differentiation, more specifically, backward divided differencing (DD), operates on some truncation of the Taylor series. This method is easy to implement by evaluating the residual vector using perturbations of the solution vector, as expressed below:

$$J_{IJ} = \frac{\partial R_I}{\partial U_J} \approx \frac{R_I(\mathbf{U} + \varepsilon \cdot e_J) - R_I(\mathbf{U})}{\varepsilon}, \quad \varepsilon \in \mathbb{R} \text{ is a small number} \tag{23}$$

where subscript $I,J = 1,2,\ldots,(Ndegr \times Netot)$ (20 for HWENO (P_1P_2)). In addition, the computational mesh needs to be divided into colored groups, which is required to confine elemental perturbation within the element itself. In another word, two face-neighboring cells cannot be in the same group. This DD-based method is very easy to implement, and the result can approach closely to the exact Jacobian matrix with a properly estimated perturbation variable ε (usually $10^{-7} - 10^{-8}$ for smooth flows).

However, there are two issues with this method. For the first, ε is in fact problem-dependent, and the method will become unstable where flow conditions change drastically, e.g., the shock waves, thus can result in a sudden breakdown of the linear solver. For the second, this method is very computationally intensive: it requires $k \times Ndegr \times Netot$ times in each Newton iteration to evaluate the residual vector, (that is $20k$ for DG (P_1) and HWENO (P_1P_2), and $50k$ for DG (P_2)), where k is the number of colored groups (k = 6–8, depending on meshes), and thus resulting in a highly expensive method.

5.3. Based on automatic differentiation

The method based on automatic differentiation (AD) is to compute the Jacobian matrix using the AD-generated source code. Automatic differentiation is a technology for automatically augmenting computer programs, including arbitrarily complex simulations, with statements for the computation of derivatives, also known as sensitivities.

By using an AD tool, e.g., TAPENADE [64] as adopted in the present work, the labor for manual coding can be significantly reduced, which otherwise can be very complicated, tedious and error-prone in the discontinuous Galerkin context if operated manually, depending on the complexity of the numerical functions. Similar to divided differencing, automatic differentiation requires only the source program C. But instead of executing C on different sets of inputs, the AD tool builds a new, augmented code C' that computes the analytical derivatives along with the source program. This new program is called the differentiated program. Each time the source program holds some value v, the differentiated program holds an additional value dv, which is the differential of v.

In the case of evaluation of the R.H.S residual vector in the present work, the reconstruction, which is before computing the face and domain integrals by using the reconstructed polynomial solution, is not included in the AD source program, due to the structure of legacy code. Thus like the method based on analytical derivation as introduced above, the source program of the underlying DG (P_1) method is provided for TAPENADE. For the inviscid part, the AD process results in an differentiated program in which the HLLC flux functions are exactly linearized. However, numerical tests conducted by Batten et al. [48] showed that the exact implicit HLLC scheme provided no further improvement than the frozen wave-speed version of this scheme for convergence in terms of computing time, which is again confirmed in the present work. For the viscous part, the AD process of face integrals results in an exact linearization of the diffusion terms, including the local lift operator of the BR2 scheme. But in the AD process of domain integrals, the global lift operator of the BR2 scheme is treated as constant. This simplification is made in order to offset the overhead of doubly computing the face-based local lift operator to formulate the cell-based global lift operator. With the attractive feature described above, the AD-based method can also largely reduce the workload for code maintenance in case the underlying numerical flux schemes are updated, as the programmer only needs to conduct the AD process with the updated source program C and use the generated code C' back in the solver code.

6. Parallelization

The compactness of the HWENO (P_1P_2) RDG method makes it ideally suited for parallel computing. In the present work, an SPMD (single program, multiple data) programming paradigm based on the MPI library is adopted to achieve parallelism, the METIS library [65] is used for the partitioning of a grid into sub-domain grids of approximately the same size. Two examples are illustrated in Fig. 2. The first one is a prismatic + hexahedral hybrid grid for flow around a circular cylinder, which is split into 16 sub-domains as shown in Fig. 2a. The second one is a tetrahedral grid for flow over a wing/pylon/finned-store configuration, which is split into 64 sub-domains as shown in Fig. 2b. The communication in parallel mode is managed by the necessary standard MPI commands like non-blocking send, nonblocking receive and wait commands. In the present work, parallelization is implemented for both the explicit and implicit methods.

7. Numerical examples

Computations on a series of well-documented test cases for the compressible inviscid and viscous flows are carried out in this section. In these test cases, the serial computations are conducted on a Dell Precision T7400 personal computer (2.98 GHz Xeon CPU with 18 GBytes memory), and the parallel computations are performed on a cluster (AMD Opteron 6128 8-core 2.0 GHz processor with 32 GBytes memory for each compute-node).

The following L^2 norm of the entropy production is used as the error measurement for the steady-state inviscid flow problems

$$\|\varepsilon\|_{L^2(\Omega)} = \sqrt{\int_\Omega \varepsilon^2 \, d\Omega} = \sqrt{\sum_{i=1}^{Nelem} \int_{\Omega_i} \varepsilon^2 \, d\Omega}$$

where the entropy production ε is defined as

$$\varepsilon = \frac{S - S_\infty}{S_\infty} = \frac{p}{p_\infty}\left(\frac{\rho_\infty}{\rho}\right)^\gamma - 1$$

Note that the entropy production, where the entropy is defined as $S = (p/\rho)^\gamma$, is a very good criterion to measure the accuracy of numerical solutions, since the flow under consideration is isentropic.

In the present test cases, the convergence speedup factor for the implicit methods is assessed according to the baseline results obtained by the three-stage TVD Runge–Kutta (TVDRK3) explicit time stepping scheme. For 2D illustration of a cluster of variables

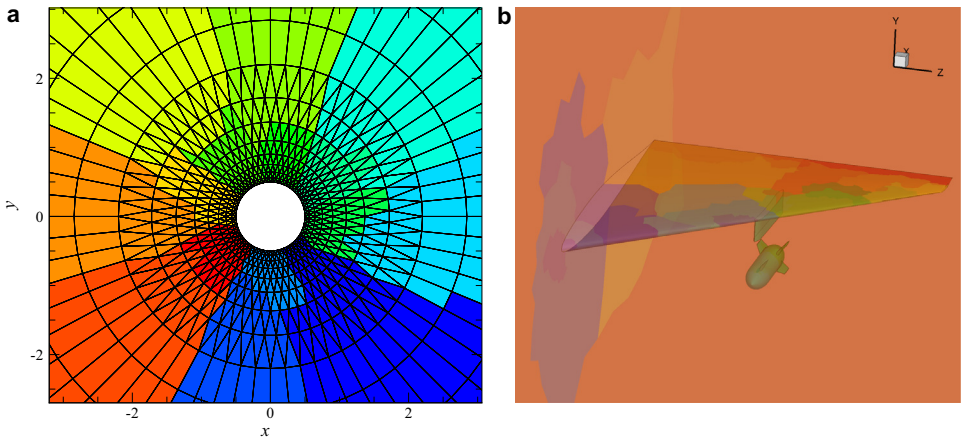

Fig. 2. Representation of domain decomposition by METIS.

on the surface of solid body, e.g., extracted surface pressure coefficients on a cut plane, the variables are computed at the two nodes of each cell face that intersect with the cut plane, and plotted by a straight line. This is the most accurate way to represent the P_1 solution, as the solution is linear on each cell face and multiple values can exist across the cell interface due to the discontinuous representation of DG solution. Some phrase abbreviations for describing a computational grid are defined as follows: Nelem, number of total elements of a grid; Ntetr, number of tetrahedral elements; Npyra, number of pyramidal elements; Npris, number of prismatic elements; Nhexa, number of hexahedral elements; Nafac, number of total boundary faces; Npoin, number of grid vertex node.

7.1. Subsonic flow past a circular cylinder

This is a well-known test case: inviscid subsonic flow past a circular cylinder at a Mach number of $M_\infty = 0.38$. This test case is chosen to test the accuracy of the WENO (P_1P_2) methods and the performance of the implicit methods on hybrid grids. This is a 3D simulation of the 2D problem. Computation is conducted on a series of four successively refined prismatic + hexahedral hybrid grids, having $(16+8)\times 4$ (Level-1), $(32+16)\times 8$ (Level-2), $(64+32)\times 16$ (Level-3) and $(128+64)\times 32$ (Level-4) cells in the x–y plane and 1 cell in the z-direction, as shown in Fig. 3a–d. Numerical solutions to this problem are computed using DG (P_1) and WENO (P_1P_2) on these grids to obtain a quantitative measurement of the

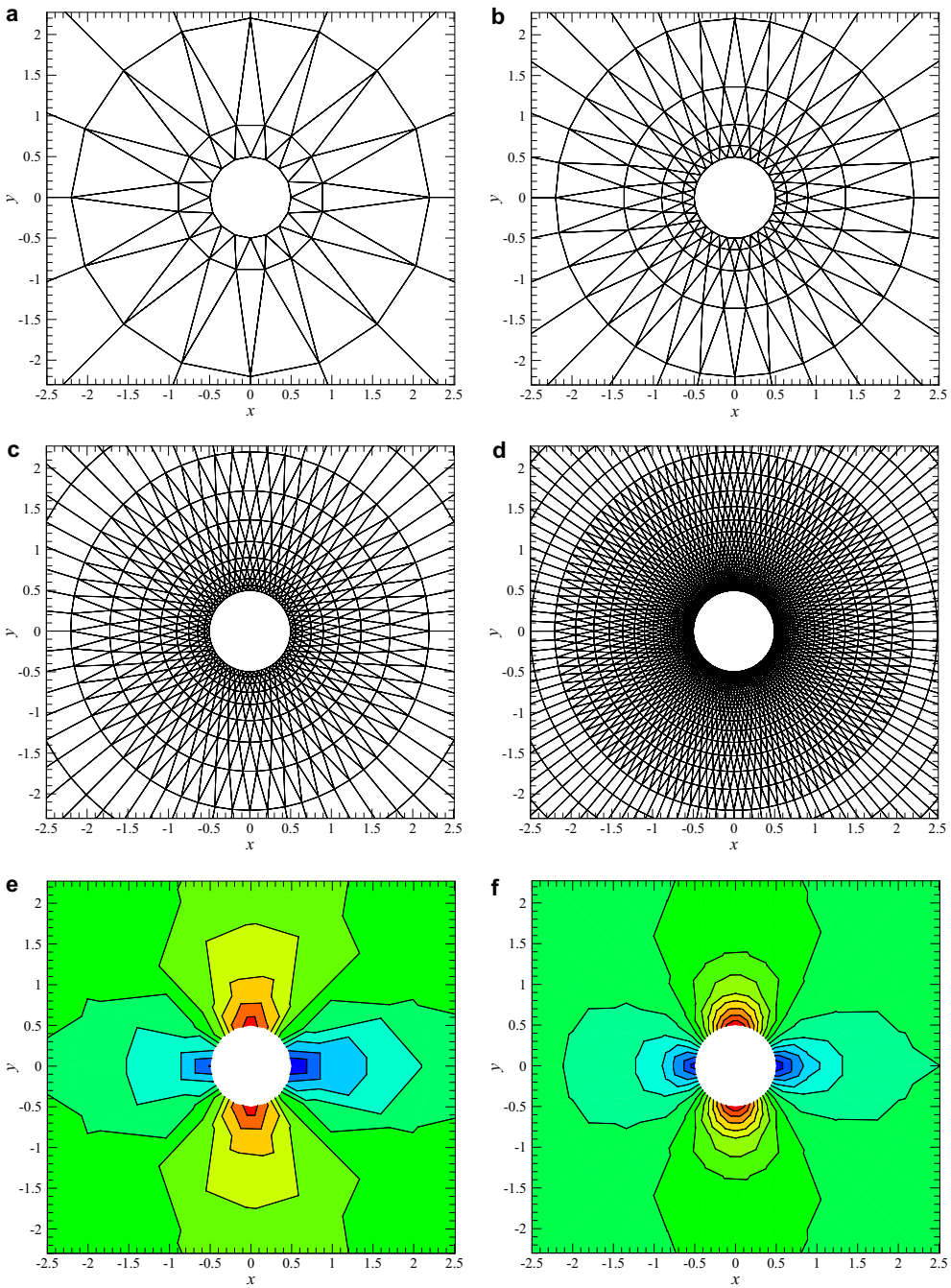

Fig. 3. (a–d) A sequence of four successively refined prismatic + hexahedral hybrid grids. Computed Mach number contours in the flow field by (e–h) DG (P_1) and (i–l) WENO (P_1P_2) solutions for inviscid subsonic flow past a circular cylinder at $M_\infty = 0.38$.

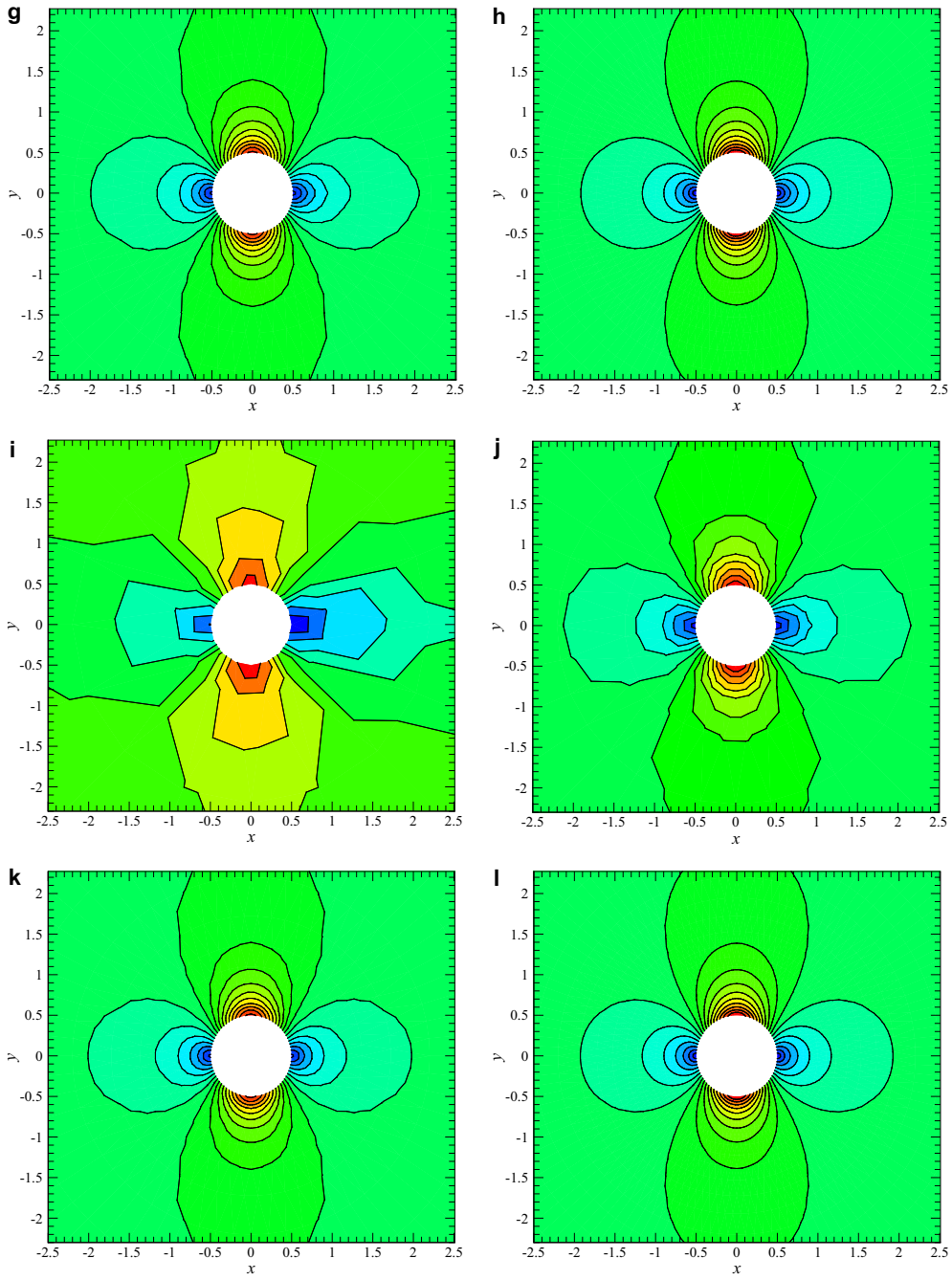

Fig. 3 (*continued*)

Table 2
The circular cylinder case on prismatic/hexahedral hybrid grids: DG (P1).

No. of cells	Cell size	L^2 error	Order (slope)
$(16+8)\times 4$	h	−0.12907E + 01	–
$(32+16)\times 8$	$h/2$	−0.19912E + 01	2.326
$(64+32)\times 16$	$h/4$	−0.27710E + 01	2.593
$(128+64)\times 32$	$h/8$	−0.35798E + 01	2.697

Table 3
The circular cylinder case on prismatic/hexahedral hybrid grids: WENO (P1P2).

No. of cells	Cell size	L^2 error	Order (slope)
$(16+8)\times 4$	h	−0.12000E + 01	–
$(32+16)\times 8$	$h/2$	−0.23306E + 01	3.762
$(64+32)\times 16$	$h/4$	−0.36842E + 01	4.500
$(128+64)\times 32$	$h/8$	−0.46498E + 01	3.219

discretization errors, which are presented in Tables 2 and 3. They show the grid size, the L^2 error of the solutions, and the piece-wise order of convergence. Fig. 3e–h, and Fig. 3i–l show the computed Mach number contours in the flow field obtained by the DG (P_1) and WENO (P_1P_2) solutions on the four grids, respectively. One can see that the WENO (P_1P_2) solutions are more accurate than the DG (P_1) solutions on the first three grids. Both the DG (P_1) and WENO (P_1P_2) solutions are virtually identical on the finest grid.

However, WENO (P_1P_2) does yield a slightly more accurate solution than DG (P_1) at the same grid resolution if we compare the L^2 errors in Tables 2 and 3. Seen from Fig. 4a, DG (P_1) and WENO (P_1P_2) each achieves an averaged slope of over 2 and 3, respectively. Fig. 4b illustrates that WENO (P_1P_2) requires significantly fewer degrees of freedom than DG (P_1) for the same accuracy. Fig. 5a and b shows a comparison of convergence histories versus time steps and CPU time on the Level-3 grid, respectively for implicit DG (P_1) and WENO (P_1P_2) using the Jacobian matrix based on analytical derivation. The implicit method converges in nearly the same number of time steps and CPU time for both DG (P_1) and WENO (P_1P_2) solutions, demonstrating the order independence of this implicit method. The performance of the implicit DG (P_1) and WENO (P_1P_2) methods is also carried out on the Level-4 grid, as displayed in Fig. 6a and b. Again, the similar order independence of the implicit method is observed.

7.2. Subsonic flow past a sphere

In this test case, an inviscid subsonic flow past a sphere at a free-stream Mach number of $M_\infty = 0.5$ is chosen to assess the accuracy of WENO (P_1P_2) and the performance of the implicit methods on 3D configurations. A sequence of three successively refined tetrahedral grids for computation are shown in Fig. 7a–c, respectively: Level-1 (Nelem = 535, Npoin = 167, Nafac = 244), Level-2 (Nelem = 2426, Npoin = 589, Nafac = 640), and Level-3 (Nelem = 16,467, Npoin = 3425, Nafac = 2372). The cell size is halved between two consecutive grids. Note that only a quarter

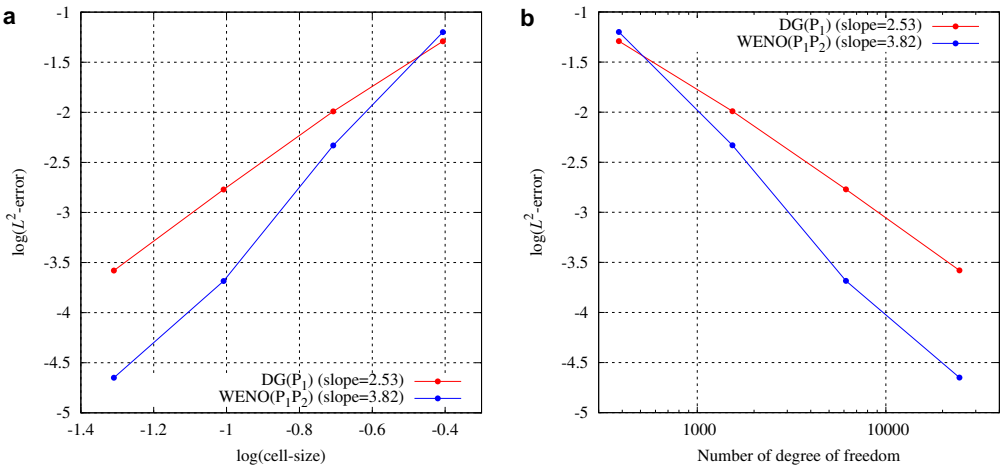

Fig. 4. L^2 errors of numerical solutions versus (a) cell size and (b) number of degree of freedom, for subsonic flow past a circular cylinder at $M_\infty = 0.38$.

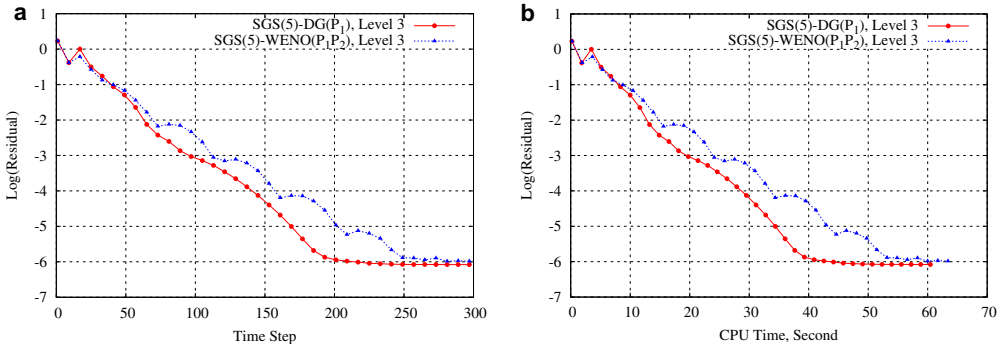

Fig. 5. Convergence history versus (a) time steps and (b) CPU time for the implicit methods on the Level-3 grid, for subsonic flow past a circular cylinder at $M_\infty = 0.38$.

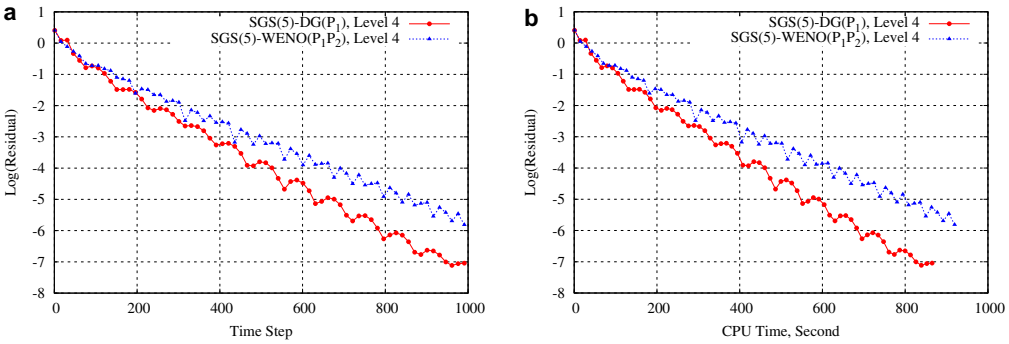

Fig. 6. Convergence history versus (a) time steps and (b) CPU time for the implicit methods on the Level-4 grid, for subsonic flow past a circular cylinder at $M_\infty = 0.38$.

of the configuration is modeled due to symmetry of the problem. Fig. 7d–f and g–i display the computed Mach number contours in the flow field by DG (P_1) and WENO (P_1P_2) solutions, respectively. Fig. 8a and b show the L^2 errors of numerical solutions versus cell size and No. DOFs respectively. The results obtained by DG (P_1) and WENO (P_1P_2), perhaps not as impressive as those shown in the previous test case likely due to the tetrahedral grid quality, do indicate that DG (P_1) and WENO (P_1P_2) exhibit a $\mathcal{O}(h^2)$ and $\mathcal{O}(h^3)$ order of convergence on smooth solutions, respectively. Fig. 9a and b displays a comparison of convergence histories versus time steps and CPU time between explicit and implicit methods, using WENO (P_1P_2) on the Level-1, Level-2 and Level-3 grids respectively. The implicit method uses the Jacobian matrix based on analytical derivation, which obtains the convergence in nearly the same number of time steps on all the three grids as shown in Fig. 9a, demonstrating the order independence of this implicit method. The implicit method is over 30 times faster than its explicit TVDRK3 counterpart as shown in Fig. 9b.

7.3. Transonic flow over the ONERA M6 wing

An inviscid transonic flow over the ONERA M6 wing at $M_\infty = 0.84$ and $\alpha = 3.06^\circ$ is considered in this case in order to assess the performance of HWENO (P_1P_2) and the implicit methods at the presence of strong discontinuities. A tetrahedral grid (Nelem = 95,266, Npoin = 18,806, Nafac = 10,579) is used in

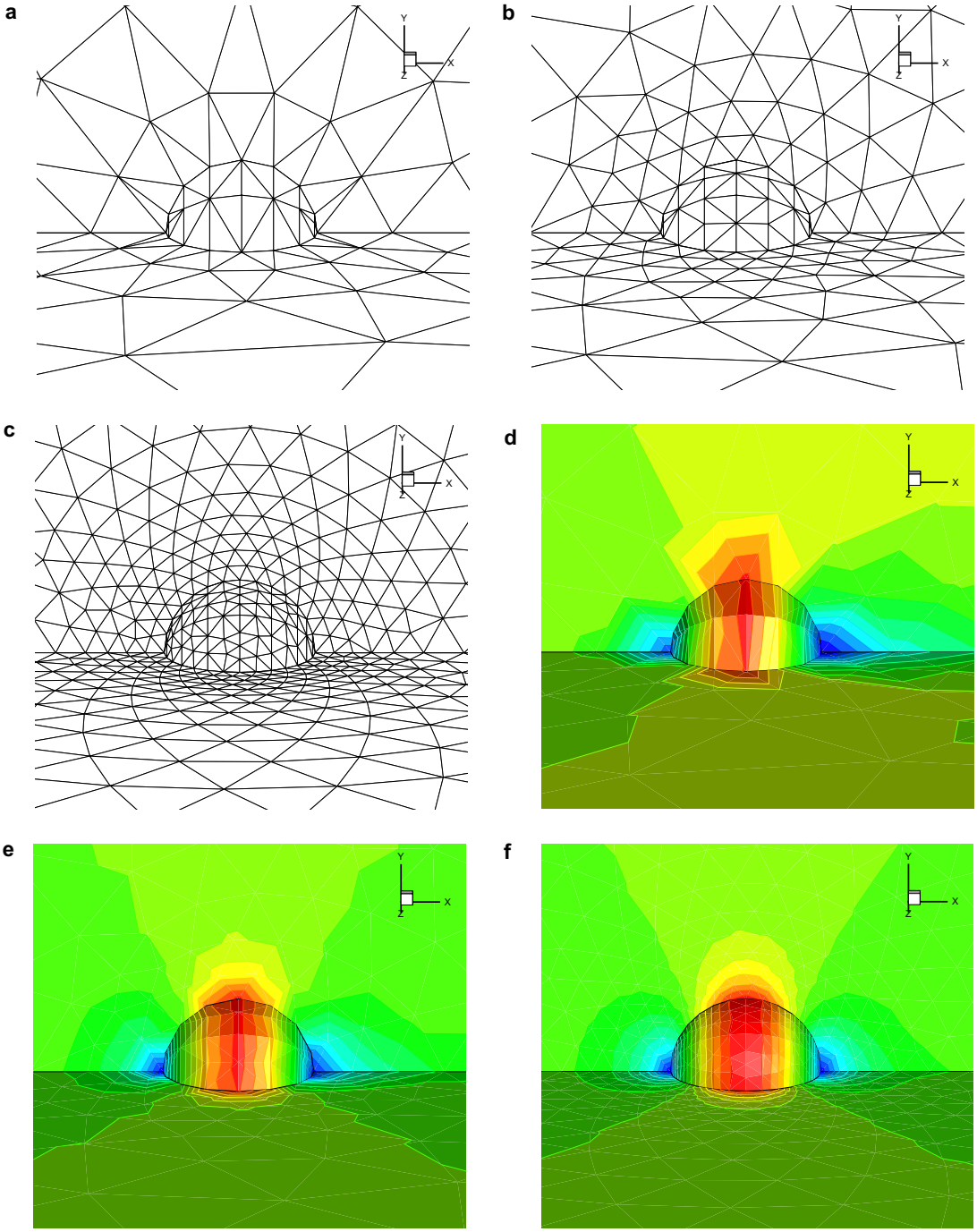

Fig. 7. (a–c) A sequence of three successively refined tetrahedral grids. Computed Mach number contours obtained by (d–f) DG (P_1) and (g–i) WENO (P_1P_2) solutions, for inviscid subsonic flow past a sphere at $M_\infty = 0.5$.

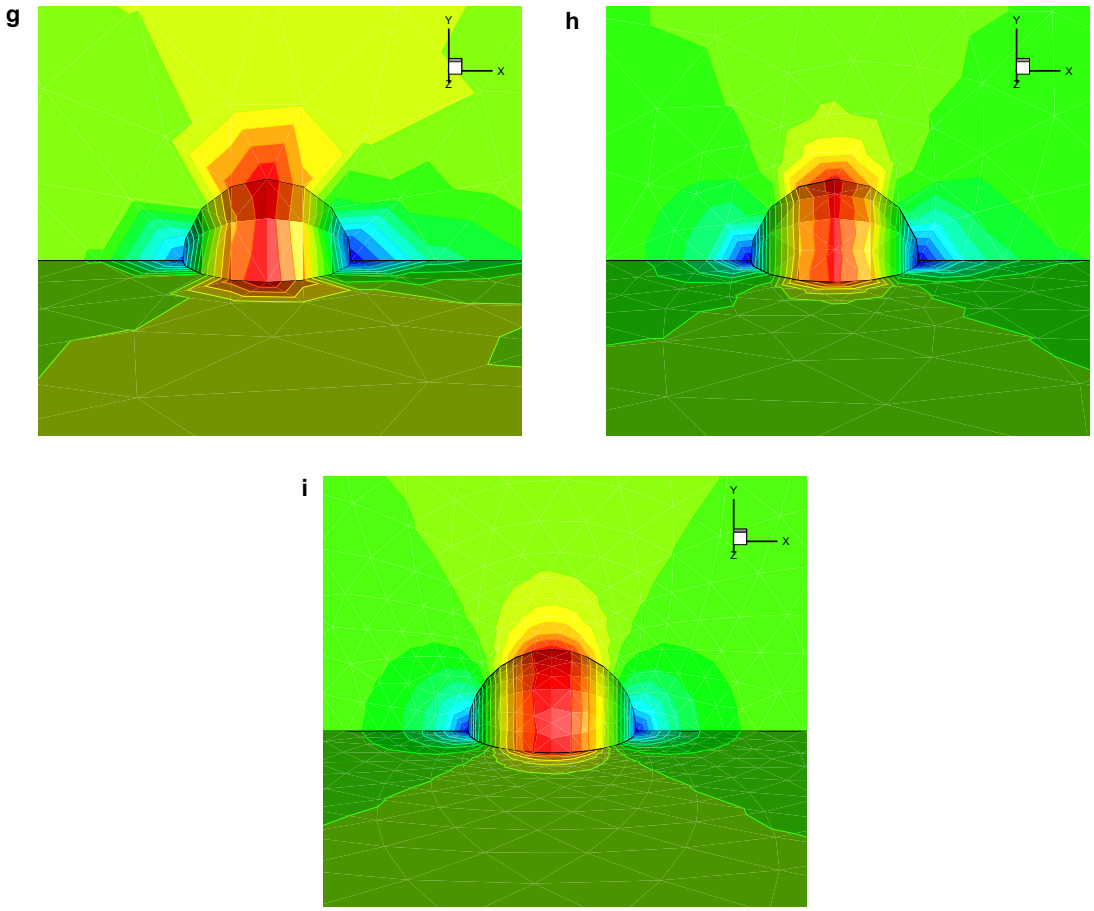

Fig. 7 (*continued*)

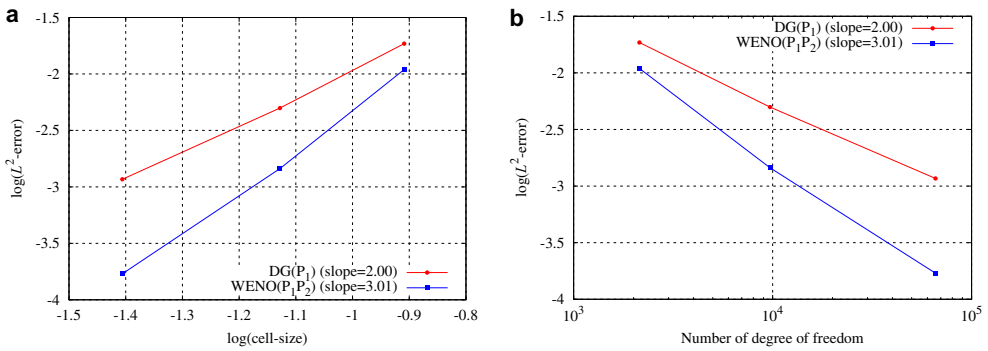

Fig. 8. L^2 errors of numerical solutions versus (a) cell size and (b) number of degree of freedom, for inviscid subsonic flow past a sphere at $M_\infty = 0.5$.

computation, as the surface triangular meshes is shown in Fig. 10a. Fig. 10b shows the computed surface pressure contours on the upper wing. The computed pressure coefficients are compared with experimental data [66] at six span-wise locations in Fig. 11a–f. The results conform closely with the experimental data, except at the root stations as shown in Fig. 11a and b, due to lack of viscous effects. The leading edge suction peak is extremely well captured in spite of the low grid resolution. The shocks are virtually captured within one cell without oscillations, clearly demonstrating the high accuracy and non-oscillatory property of our HWENO (P_1P_2) reconstruction scheme. Fig. 12a and b displays the convergence histories versus time steps and CPU time respectively, obtained by the implicit method, in which the Jacobian matrix is based on analytical derivation. This is a one-minute job by taking advantage of parallel computing on 128 CPU processors! Besides, note that the residual level would not further drop down after a decrease of 3 or 4 orders of magnitude in this test case, although the flow field has reached the steady state. This phenomenon is sometimes called "convergence stall", which could be observed for problems with strong discontinuities or sharp edges, especially when nonlinear approaches like WENO schemes are used.

7.4. Transonic flow over the wing/pylon/finned-store configuration

An inviscid transonic flow over the wing/pylon/finned-store configuration at a free-stream Mach number of $M_\infty = 0.95$ and attack angle of $\alpha = 0^\circ$ is considered in this test case, in order to

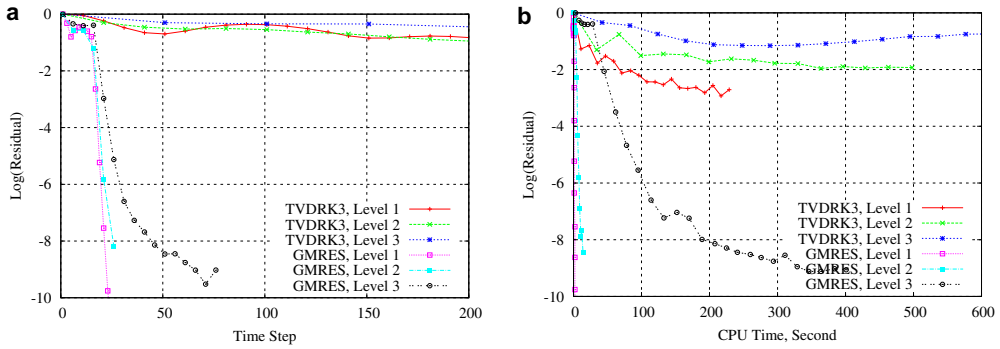

Fig. 9. Convergence history versus (a) time steps and (b) CPU time between the explicit and implicit methods, using WENO (P_1P_2) respectively on the three grids, for inviscid subsonic flow past a sphere at $M_\infty = 0.5$.

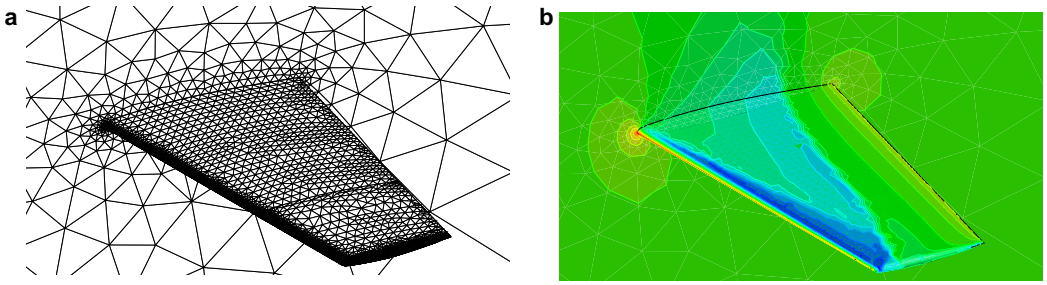

Fig. 10. (a) Surface meshes of the tetrahedral grid. (b) Computed pressure contours in the flow field obtained by HWENO (P_1P_2) solutions, for transonic flow over the ONERA M6 wing at $M_\infty = 0.84, \alpha = 3.06^\circ$.

assess the performance of HWENO (P_1P_2) and the implicit methods for complex geometric configurations. The detailed description of this configuration is reported by Ref. [67]. The configuration consists of a clipped delta wing with a 45° sweep comprised from a constant NACA 64010 symmetric airfoil section. The wing has a root chord of 15 inches, a semi-span of 13 inches, and a taper ratio of 0.134. The pylon is located at the midspan station and has a cross-section characterized by a flat plate closed at the leading and trailing edges by a symmetrical olive shape. The width of the pylon is 0.294 inches. The four fins on the store are defined by a constant NACA 0008 airfoil section with a leading-edge sweep of 45° and a truncated tip. Computation is conducted on a tetrahedral grid (Nelem = 319,134, Npoin = 147,289, Nafac = 28,738), as the surface meshes are shown in Fig. 13a and b. Fig. 13c and d show the computed pressure contours on the upper and lower wing surfaces, respectively. The computed pressure coefficient distributions are compared with experimental data at two span-wise stations in Fig. 14a and b, respectively. The comparison with experimental data is excellent on both upper and lower surfaces up to 70% chord. As expected from the Euler solution, the computation predicts a shock location that is downstream of that measured by the experiment due to the lack of viscous effect. Again, our HWENO (P_1P_2) method captures the shock waves very sharply within one cell without visible under- and over-shoots. Fig. 15a and b show the convergence histories of versus time steps and CPU time respectively, obtained by the implicit method, in which the Jacobian matrix is based on analytical derivation. By taking advantage of parallel computing on 128 CPU processors, this test case becomes simply a five-minute job, demonstrating the efficiency of the parallel, implicit methods developed in the present work.

7.5. Laminar flow past a flat plate

The laminar boundary layer over an adiabatic flat plate at a free-stream Mach number of $M_\infty = 0.5$ and a Reynolds number of Re = 100,000 based on the free-stream velocity and the length of the flat plate is considered in this test case. This problem is chosen to assess (1) the accuracy of the WENO (P_1P_2) scheme for the discretization of the viscous and heat fluxes in the Navier–Stokes equations, as the classical Blasius solution can be used to measure the accuracy of the numerical solution, (2) performance of the implicit method for viscous flows. The computational domain is bounded from -0.5 to 1.0 along the x-direction, from 0 to 1.0 along the y-direction, and from 0 to 0.1 along the z-direction, and the no-slip surface starts at point $(0, 0, z)$ and extends to $(1, 0, z)$. A slip condition is prescribed along the bottom side of the domain for $x \in [-0.5, 0]$ with $v = 0$. Symmetry conditions are prescribed for the front and back boundary with $w = 0$. The characteristic boundary is prescribed to the left side ($x = -0.5$), top side ($y = 1$), and right side ($x = 1$).

Firstly, computation is conducted on a tetrahedral grid (Ntetr = 47,536) as shown in Fig. 16a. Fig. 16b renders a closer view at the near-wall region, which depicts the sparsity of the grid resolution in the boundary layer. Fig. 16c compares the computed c_f distributions obtained by DG (P_1) and WENO (P_1P_2) solutions with the analytical solution. They are highly accurate on this grid. The difference is only discernible if one takes a zoom-in observation, e.g., for $x \in [0.5, 0.6]$ in Fig. 16d, where WENO (P_1P_2) presents a slightly better prediction than DG (P_1). Fig. 16e and f compare the computed velocity profiles with the analytical solutions at the three downstream locations, where both the DG (P_1) and WENO (P_1P_2) solutions exhibit a consistent convergence in the flow field. Fig. 17a and b display the convergence histories versus time steps and CPU time respectively for both the explicit and implicit methods. The GMRES method is chosen as the linear solver in the implicit methods. For implicit DG (P_1), the Jacobian matrix based on automatic differentiation (denoted as GMRES-AD-DG (P_1)) provides the fastest convergence of the three methods. The one based on divided differencing (denoted as GMRES-DD-DG (P_1)), although requires much fewer time steps than the one based on analytical

differentiation (denoted as MD-GMRES-DG (P_1)), turns out to be the slowest in terms of CPU time among the three methods. For implicit WENO (P_1P_2), only the one based on AD (denoted as GMRES-AD-WENO (P_1P_2)) achieved an effective convergence in this test case, indicating the high robustness to use the AD technique to linearize the viscous terms in the WENO (P_1P_2) RDG method. Overall, the implicit methods are 30 times faster than their TVDRK3 counterpart, demonstrating the superior advantage of using the developed implicit methods for viscous flow problems.

Secondly, computation is conducted on a prismatic + hexahedral hybrid grid (Npris = 977, Nhexa = 358) as shown in Fig. 18a. A zoom-in observation in Fig. 18b shows that the boundary layer region mainly consists of hexahedral elements, except at the leading edge region of the plate, where prismatic elements are filled. Fig. 18c shows the comparison of the computed c_f distributions obtained by DG (P_1) and WENO (P_1P_2) solutions with the analytical solution. A zoom-in observation for $x \in [0.02, 0.1]$ in Fig. 18d clearly demonstrates a more accurate prediction presented by WENO (P_1P_2) than DG (P_1). Fig. 18e and f compare the computed velocity profiles with the analytical solutions at three downstream locations, where both the DG (P_1) and WENO (P_1P_2) solutions again demonstrate a highly consistent convergence in the flow field. Fig. 19a and b show the convergence histories versus time steps and CPU time respectively for the implicit DG (P_1) and WENO (P_1P_2) methods (denoted as SGS (5)-AD-DG (P_1) and SGS (5)-AD-WENO (P_1P_2)), in which the SGS (5) method is chosen as the linear

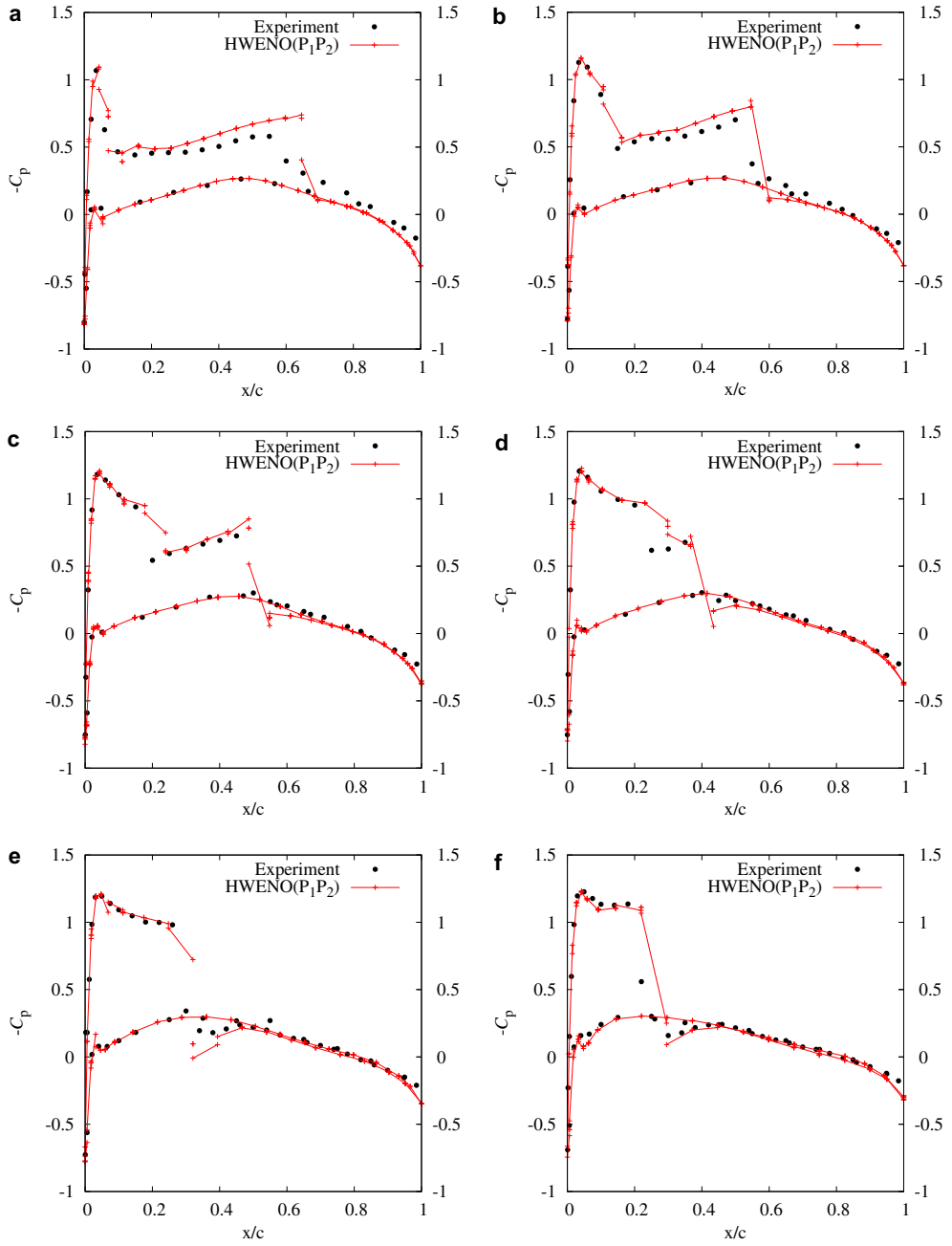

Fig. 11. Plot of pressure coefficient distributions at six span-wise locations for transonic flow over the ONERA M6 wing at $M_\infty = 0.84, \alpha = 3.06^\circ$: (a) $\eta = 0.20$, (b) $\eta = 0.44$, (c) $\eta = 0.65$, (d) $\eta = 0.80$, (e) $\eta = 0.90$, (f) $\eta = 0.95$.

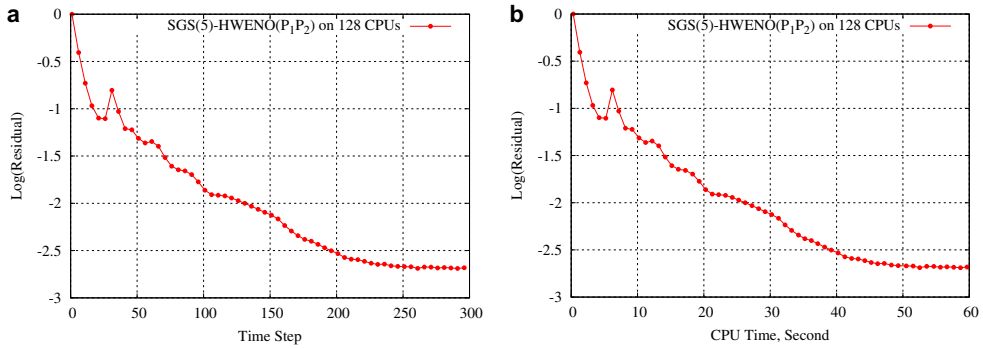

Fig. 12. Convergence history versus (a) time steps and (b) CPU time for the implicit HWENO (P_1P_2) method on 128 CPU processors, for transonic flow over the ONERA M6 wing at $M_\infty = 0.84, \alpha = 3.06^\circ$.

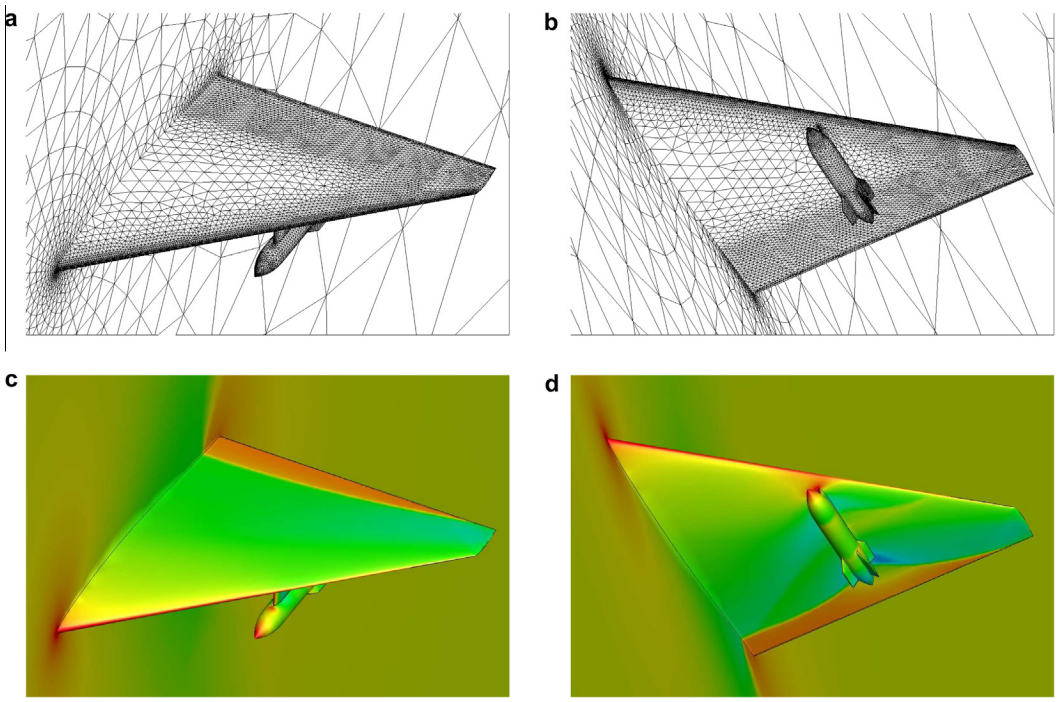

Fig. 13. (a–b) The surface meshes of a tetrahedral grid. (c–d) Computed pressure contours on the unstructured surface mesh of a tetrahedral grid obtained by the HWENO (P_1P_2) solutions for a transonic flow over a wing/pylon/finned-store configuration at $M_\infty = 0.95, \alpha = 0^\circ$.

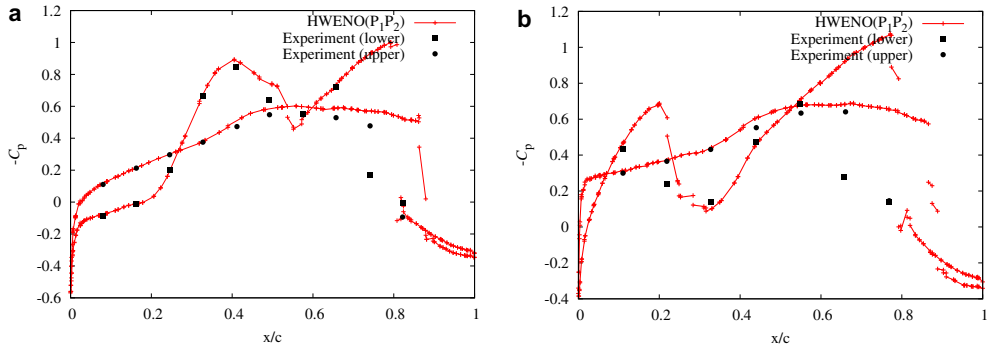

Fig. 14. Comparison of the computed pressure coefficient distributions with experimental data at two span-wise locations for a transonic flow over a wing/pylon/finned-store configuration at $M_\infty = 0.95, \alpha = 0^\circ$: (a) $\eta = 0.4077$, (b) $\eta = 0.5900$.

solver, and the Jacobian matrix is based on automatic differentiation. Compared with the implicit DG (P_1), the implicit WENO (P_1P_2) obtains the convergence without a significant increase in terms of both time steps and CPU time, demonstrating the order independence of the implicit method for computing viscous flows on hybrid grids.

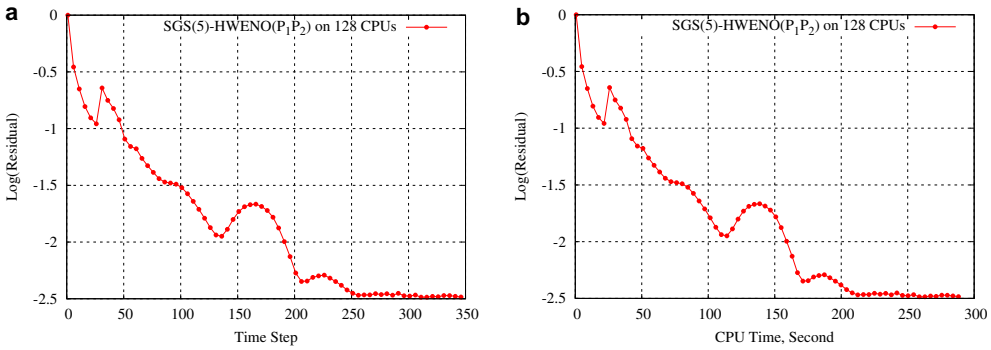

Fig. 15. Convergence history versus (a) time steps and (b) CPU time for the implicit HWENO (P_1P_2) method on 128 CPU processors, for a transonic flow over a wing/pylon/finned-store configuration at $M_\infty = 0.95, \alpha = 0^\circ$.

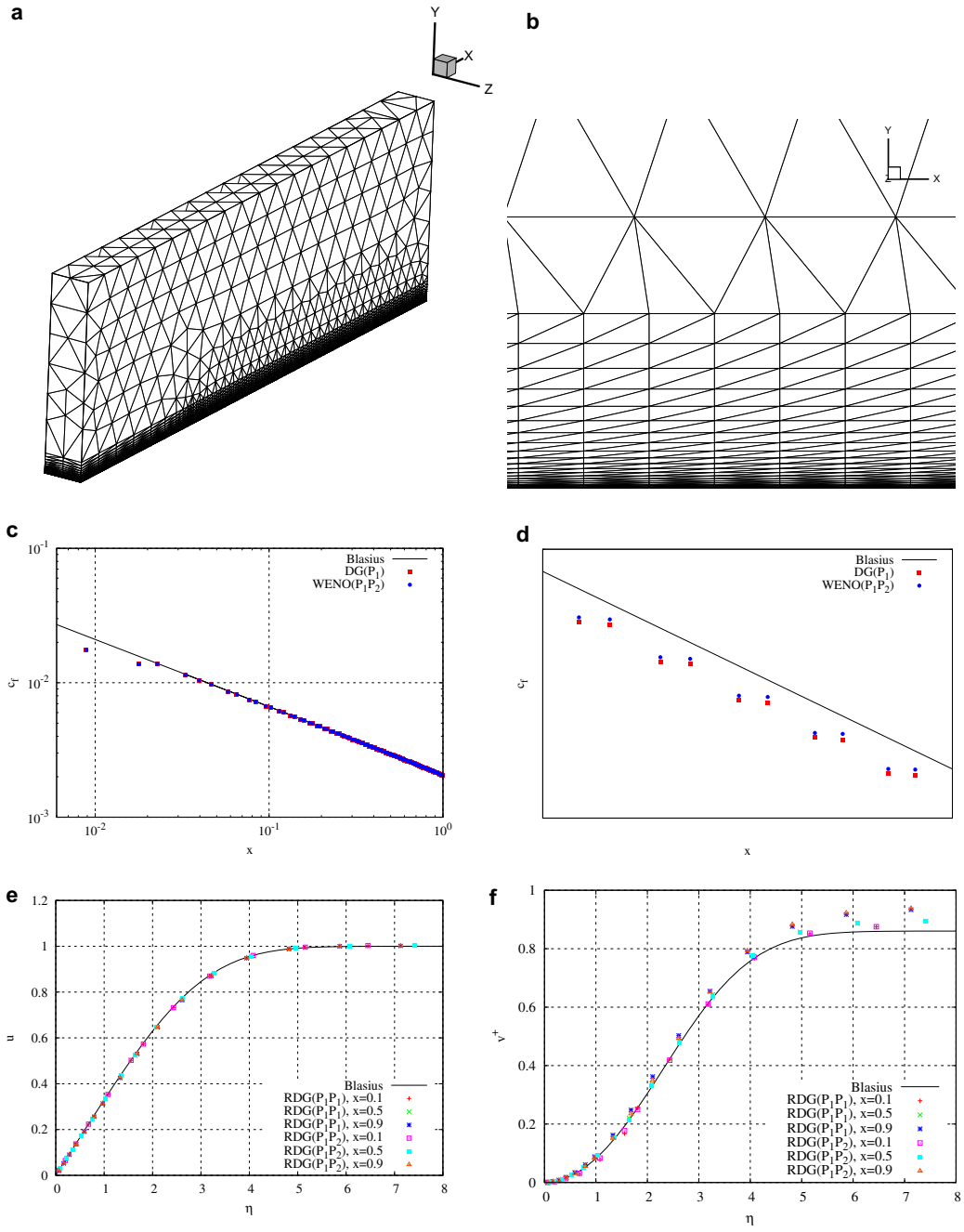

Fig. 16. Plot of the tetrahedral grids for laminar flow past a flat plate at $Re = 100,000$: (a) global domain; (b) boundary layer region. (c) Logarithmic plot of the computed skin friction coefficient c_f distribution along the flat plate $x \in [0, 1]$. (d) c_f distribution at $x \in [0.5, 0.6]$. (e) x-velocity u versus η. (f) Scaled y-velocity v^+ versus η.

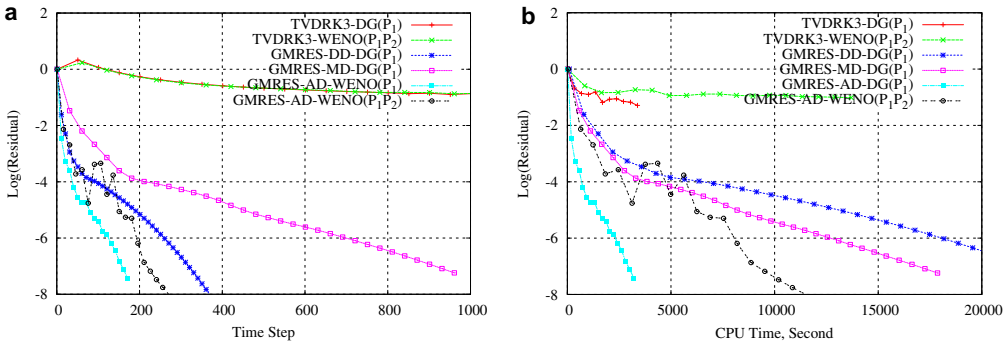

Fig. 17. Convergence histories versus (a) time steps and (b) CPU time using the explicit and implicit methods on the tetrahedral grid, for laminar flow past a flat plate at Re = 100,000.

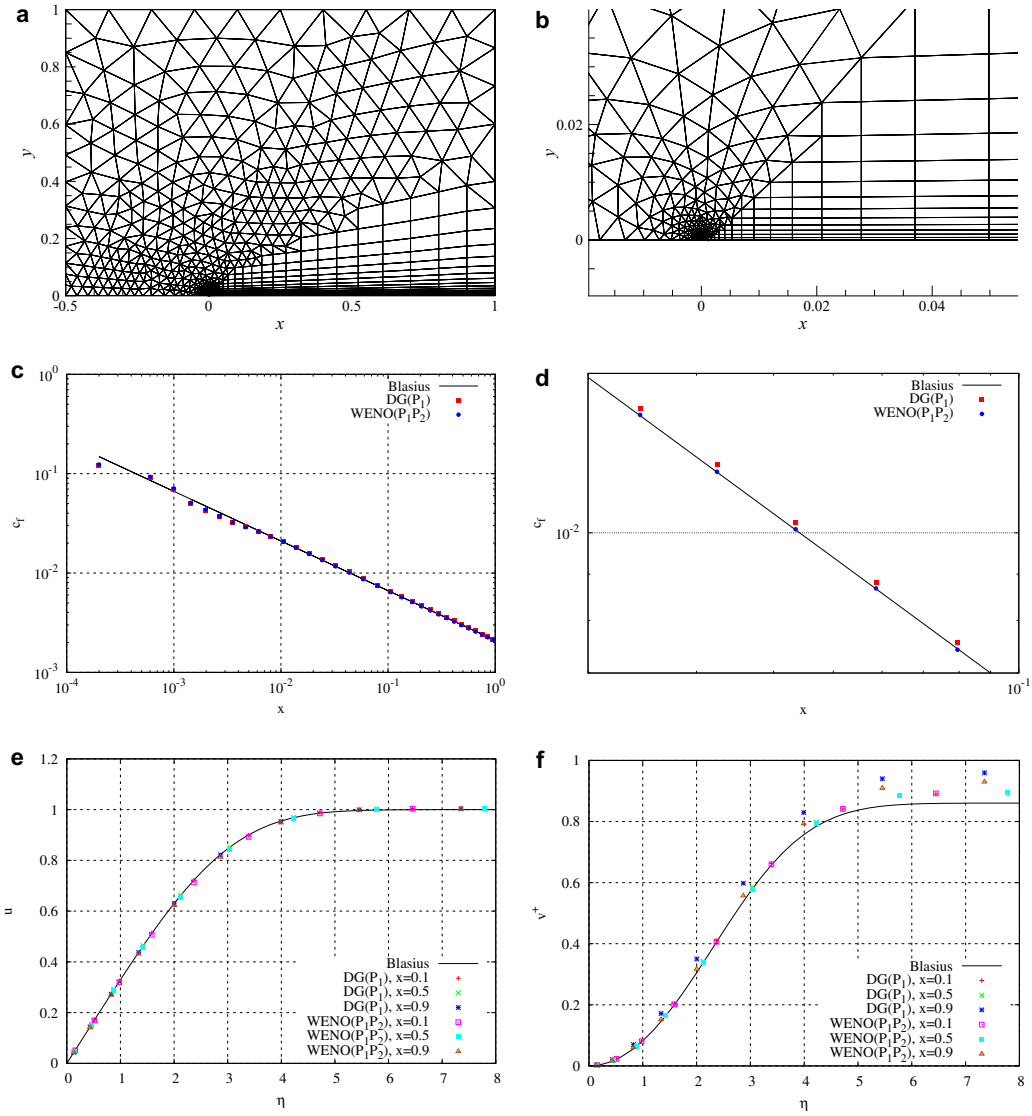

Fig. 18. Plot of the prismatic + hexahedral hybrid grids for laminar flow past a flat plate at Re = 100,000: (a) global domain; (b) boundary layer region; (c) logarithmic plot of the computed skin friction coefficient c_f distribution along the flat plate $x \in [0, 1]$; (d) c_f distribution at $x \in [0.02, 0.1]$; (e) x-velocity u versus η; (f) scaled y-velocity v^+ versus η.

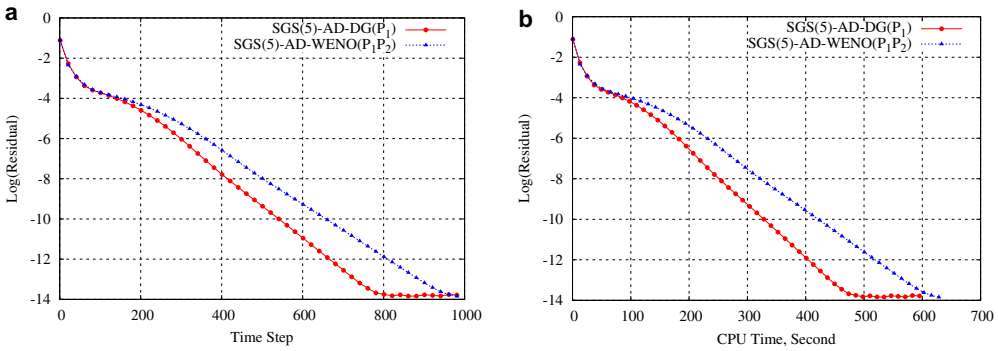

Fig. 19. Convergence histories versus (a) time steps and (b) CPU time using the implicit methods on prismatic + hexahedral hybrid grid, for laminar flow past a flat plate at Re = 100,000.

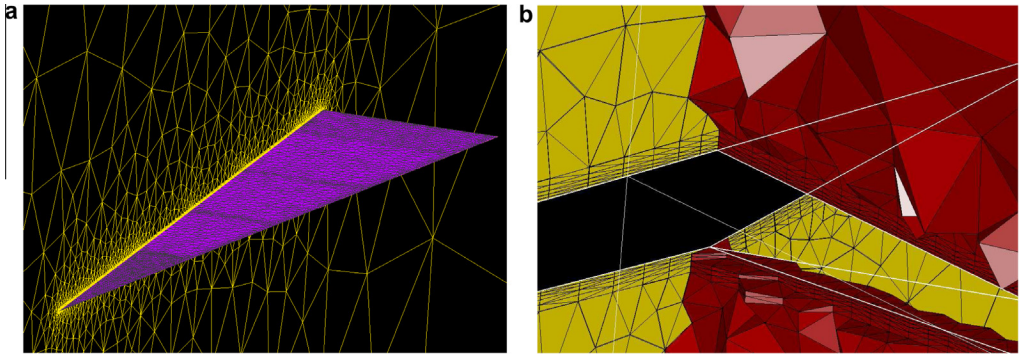

Fig. 20. A tetrahedral grid for subsonic flow over a sharp-edged slender delta wing at $M_\infty = 0.3, \alpha = 20.5^\circ$ and $Re = 0.95 \times 10^6$: (a) triangular surface meshes; (b) extracted meshes on sliced plane of $x/L = 0.9$.

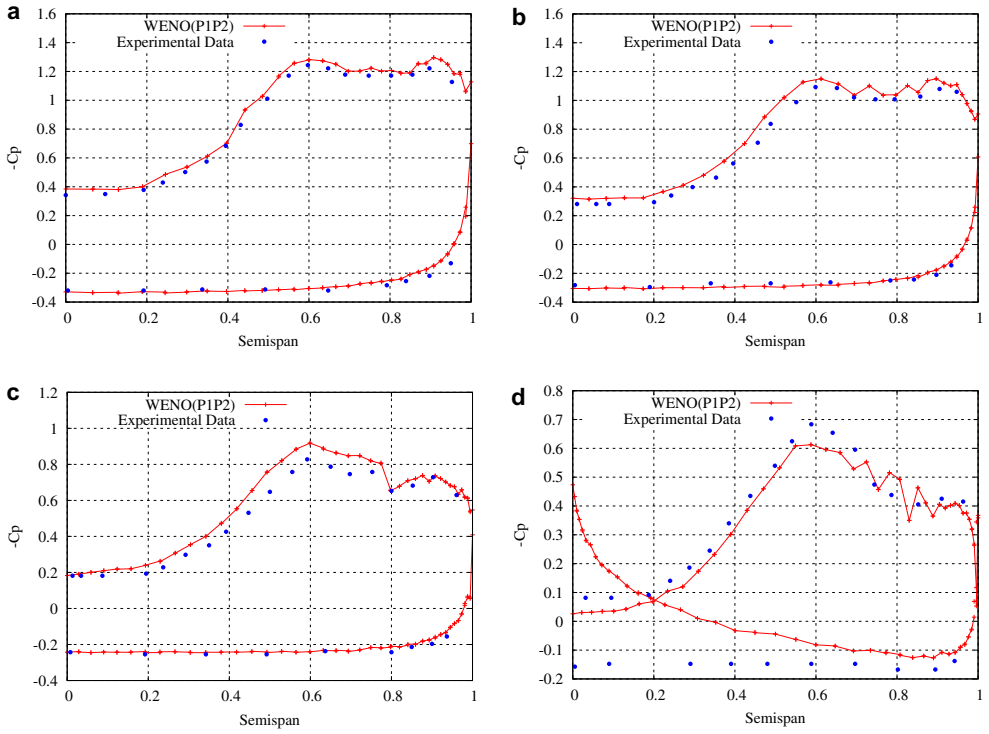

Fig. 21. Surface pressure coefficient distributions obtained by the WENO (P_1P_2) solutions at four typical downstream intersections for subsonic flow over a delta wing at $M_\infty = 0.3, \alpha = 20.5^\circ$ and $Re = 0.95 \times 10^6$: (a) $x/L = 0.3$, (b) $x/L = 0.5$, (b) $x/L = 0.7$, (c) $x/L = 0.9$.

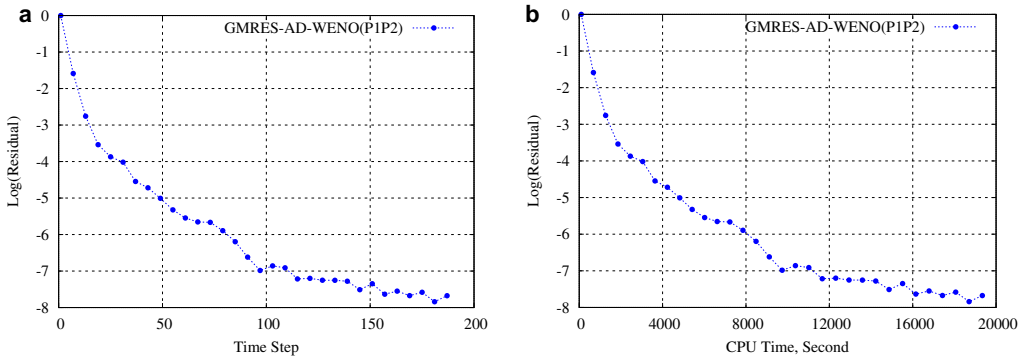

Fig. 22. Convergence histories versus (a) time steps and (b) CPU time, using the implicit methods for subsonic flow over a delta wing at $M_\infty = 0.3, \alpha = 20.5^\circ$ and $Re = 0.95 \times 10^6$.

7.6. Subsonic flow around a delta wing at $Re = 0.95 \times 10^6$

A subsonic flow around a sharp-edged slender delta wing at a free-stream Mach number of $M_\infty = 0.3$, angle of attack $\alpha = 20.5^\circ$, and Reynolds number of $Re = 0.95 \times 10^6$ based on a mean cord length of 1 is presented in this test case in order to assess the performance of our implicit WENO (P_1P_2) RDG method for computing high Reynolds number flows, as the experimental data is available for the geometric configuration [68] considered. The aspect ratio of this delta wing is equal to 1, which corresponds to a sweep angle of 75.9638°. The upper surface of the wing is flat and the cross section is triangular ahead of $x/L = 0.9$, with the maximum thickness $0.021L$. The cross section downstream of $x/L = 0.9$ is trapezoidal, and the trailing edge is sharp. A tetrahedral grid (Nelem = 316,139, Npoin = 58,322, Nafac = 21,712) is used in computation, as the triangular meshes of the wing surface are displayed in Fig. 20a, where the no-slip and adiabatic boundary conditions are prescribed. Fig. 20b displays a sliced-plane of $x/L = 0.9$, where one can see the highly stretched anisotropic elements piled on the wing surface. The height of the elements on the first layer is 0.5×10^{-3}, with a growth rate of 1.3 normal to the surface and 5 layers in total. The rest of the domain is filled with isotropic elements. Computation is initialized with constant free-stream values in the entire domain. The flow filed is assumed to have reached the steady state using the implicit WENO (P_1P_2) method (Jacobian matrix based on automatic differentiation) after a decrease of 8 orders of magnitude for the global density residual in only 191 time steps as shown in Fig. 22a. The convergence history versus CPU time is shown in Fig. 22b. Fig. 21a–d show the computed surface pressure coefficient distributions compared with the experimental data at four stations along the chord of the geometry. At the forward and middle stations as shown in Fig. 21a–c, the numerical results agree well with the experimental data, However the pressure coefficients are significantly over-predicted on the lower wing surface at station $x/L = 0.9$ as shown in Fig. 21d. Similar results were previously reported in Ref. [69,63]. The reason for the disagreement is not clearly known, but it was surmised that the disagreement might be caused by the presence of the pressure tubing that exits from the model in the lower surface trailing-edge region [69].

8. Conclusion

A set of implicit methods are developed for the third-order Hierarchical WENO (P_1P_2) reconstruction based discontinuous Galerkin method for the solution of compressible flows on 3D hybrid grids. These implicit methods compute the Jacobian matrix arising from Newton linearization based on the underlying P_1 element approximation. In particular, three approaches: analytical derivation, divided differencing, and automatic differentiation are designed to construct the Jacobian matrix respectively, where the AD-based approach have shown the best robustness. Furthermore, these implicit methods are verified to compute the flows with discontinuities without any difficulty. The advantages of the developed implicit method lie in two aspects. On one side, this method can achieve the third-order accuracy, adding one order to the underlying implicit DG (P_1) method without significant extra cost in computing time and memory requirement. On the other, this method only requires less than 1/6 of the memory space of the implicit DG (P_2) method to achieve the same order of accuracy, resulting in a fast as well as low storage method that can be efficiently used to accelerate the convergence. An SPMD (single program, multiple data) programming paradigm based on MPI has been employed to achieve parallelism. The numerical results indicate that the developed implicit methods are orders of magnitude faster than their explicit counterpart. The performance of parallel computing shows that using these implicit methods, the HWENO (P_1P_2) RDG method provides a viable and attractive DG solution for complicated flows of practical importance.

References

[1] Cockburn B, Hou S, Shu CW. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case. J Math Phys 1990;55:545–81.
[2] Cockburn B, Shu CW. The Runge-Kutta discontinuous Galerkin method for conservation laws V: multidimensional system. J Comput Phys 1998;141:199–224.
[3] Bassi F, Rebay S. High-order accurate discontinuous finite element solution of the 2D Euler equations. J Comput Phys 1997;138:251–85.
[4] Bassi F, Rebay S. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. J Comput Phys 1997;131(2):267–79.
[5] Bassi F, Rebay S. Discontinuous Galerkin solution of the reynolds-averaged Navier–Stokes and κ–ω turbulence model equations. Comput Fluids 2005;34(4–5):507–40.
[6] Cockburn B, Shu C. The local discontinuous Galerkin method for time-dependent convection-diffusion system. SIAM J Numer Anal 1998;35(6):2440–63.
[7] Peraire J, Persson PO. The compact discontinuous Galerkin method for elliptic problems. SIAM J Sci Comput 2008;30(4):1806–24.
[8] Bassi F, Rebay S. A high order discontinuous Galerkin method for compressible turbulent flow. In: Cockburn B, Karniadakis GE, Shu CW, editors. Discontinuous Galerkin methods, theory, computation, and applications. Lecture notes in computational science and engineering, vol. 11; 2000.
[9] Klaij CM, van der Vegt JJW, van der Ven H. Space-time discontinuous Galerkin method for the compressible Navier–Stokes equations. J Comput Phys 2006;217(2):589–611.
[10] Qiu J, Shu CW. Runge-Kutta discontinuous Galerkin method using WENO limiters. SIAM J Sci Comput 2005;26(3):907–29.
[11] Qiu J, Shu CW. Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method: one-dimensional case. J Comput Phys 2004;193(1):115–35.

[12] Qiu J, Shu CW. Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method II: two dimensional case. J Comput Phys 2005;34(6):642–63.
[13] Luo H, Baum JD, Löhner R. A discontinuous Galerkin method using Taylor basis for compressible flows on arbitrary grids. J Comput Phys 2008;227(20):8875–93.
[14] Luo H, Luo L, Xu K. A discontinuous Galerkin method based on a BGK scheme for the Navier–Stokes equations on arbitrary grids. Adv Appl Math Mech 2009;1(3):301–18.
[15] van Leer B, Nomura S. Discontinuous Galerkin method for diffusion. AIAA Paper 2005:2005–5108.
[16] van Leer B, Lo M. A discontinuous Galerkin method for diffusion based on recovery. AIAA Paper 2007:2007–4083.
[17] Krivodonova L, Berger M. High-order accurate implementation of solid wall boundary conditions in curved geometries. J Comput Phys 2006;211(2):492–512.
[18] Ainsworth M. Dispersive and dissipative behaviour of high order discontinuous Galerkin finite element methods. J Comput Phys 2004;198(1):106–30.
[19] Arnold DN. An interior penalty finite element method with discontinuous elements. SIAM J Numer Anal 1982;19(4):742–60.
[20] Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified analysis of discontinuous Galerkin methods for elliptic problems. SIAM J Numer Anal 2002;39(5):1749–79.
[21] Collis SS. Discontinuous Galerkin methods for turbulence simulation. In: Proceedings of the 2002 center for turbulence research summer program. Citeseer; 2002. p. 155–67.
[22] Hartmann R. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier–Stokes equations. Int J Numer Methods Fluids 2006;51(9–10):1131–56.
[23] Hartmann R, Houston P. Symmetric interior penalty DG methods for the compressible Navier–Stokes equations II: goal-oriented a posteriori error estimation; 2005.
[24] Oliver TA, Fidkowski KJ, Darmofal DL. Multigrid solution for high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. Springer; 2006.
[25] Klaij CM, Van der Vegt JJW, Van der Ven H. Pseudo-time stepping methods for space-time discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. J Comput Phys 2006;219(2):622–43.
[26] Persson PO, Peraire J. Sub-cell shock capturing for discontinuous Galerkin methods. AIAA Paper 2006;112:2006.
[27] Reed W, Hill T. Triangular mesh methods for the neutron transport equation. Los Alamos scientific laboratory report. LA-UR–73-479; 1973.
[28] Luo H, Xia Y, Spiegel S, Nourgaliev R, Jiang Z. A reconstructed discontinuous Galerkin method based on a hierarchical WENO reconstruction for compressible flows on tetrahedral grids. J Comput Phys 2013;236:477–92.
[29] Dumbser M, Zanotti O. A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes. J Comput Phys 2008;227(18):8209–53.
[30] Dumbser M. Arbitrary high order PNPM schemes on unstructured meshes for the compressible Navier–Stokes equations. Comput Fluids 2010;39(1):60–76.
[31] Dumbser M, Zanotti O. Very high order P_NP_M schemes on unstructured meshes for the resistive relativistic MHD equations. J Comput Phys 2009;228(18):6991–7006.
[32] Balsara D, Altmann C, Munz C, Dumbser M. A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG + HWENO schemes. J Comput Phys 2007;226(1):586–620.
[33] Luo H, Luo L, Nourgaliev R, Mousseau V, Dinh N. A reconstructed discontinuous Galerkin method for the compressible Navier–Stokes equations on arbitrary grids. J Comput Phys 2010;229(19):6961–78.
[34] Luo H, Xia Y, Nourgaliev R, Cai C. A class of reconstructed discontinuous Galerkin methods for the compressible flows on arbitrary grids. AIAA Paper 2011:2011–199.
[35] Luo H, Luo L, Ali A, Nourgaliev R, Cai C. A parallel, reconstructed discontinuous Galerkin method for the compressible flows on arbitrary grids. Commun Comput Phys 2011;9(2):363–89.
[36] Luo H, Xia Y, Li S, Nourgaliev R. A hermite WENO reconstruction-based discontinuous Galerkin method for the Euler equations on tetrahedral grids. J Comput Phys 2012;231(16):5489–503.
[37] Xia Y, Frisbey M, Luo H, Nourgaliev R. A WENO reconstruction-based discontinuous Galerkin method for compressible flows on hybrid grids. AIAA Paper 2013. 2013-0516.
[38] Zhang L, Wei L, He L, Deng X, Zhang H. A class of hybrid DG/FV methods for conservation laws I: basic formulation and one-dimensional systems. J Comput Phys 2012;231(4):1081–103.
[39] Zhang LP, Liu W, He LX, Deng XG, Zhang HX. A class of hybrid DG/FV methods for conservation laws II: two dimensional cases. J Comput Phys 2012;231(4):1104–20.
[40] Bassi F, Rebay S. GMRES discontinuous Galerkin solution of the compressible Navier–Stokes equations. In: Cockburn B, Karniadakis GE, Shu CW, editors. Discontinuous Galerkin methods, theory, computation, and applications. Lecture notes in computational science and engineering, vol. 11; 2000b. p. 197–208.
[41] Rasetarinera P, Hussaini MY. An efficient implicit discontinuous spectral Galerkin method. J Comput Phys 2001;172:718–38.
[42] Fidkowski KJ, Oliver TA, Lu J, Darmofal DL. *p*-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. J Comput Phys 2005;207(1):92–113.
[43] Yasue K, Furudate M, Ohnishi N, Sawada K. Implicit discontinuous Galerkin method for RANS simulation utilizing pointwise relaxation algorithm. Commun Comput Phys 2010;7(3):510–33.
[44] Crivellini A, Bassi F. An implicit matrix-free discontinuous Galerkin solver for viscous and turbulent aerodynamic simulations. Comput Fluids 2011;50(1):81–93.
[45] Xia Y, Luo H, Nourgaliev R. An implicit method for a reconstructed discontinuous Galerkin method on tetrahedron grids. AIAA Paper 2012:2012–834.
[46] Xia Y, Nourgaliev R. An implicit hermite WENO reconstruction-based discontinuous Galerkin method on tetrahedral grids. In: 7th international conference on computational fluid dynamics. ICCFD7-4205; 2012.
[47] Xia Y, Luo H, Nourgaliev R. An implicit reconstructed discontinuous Galerkin method based on automatic differentiation for the Navier–Stokes equations on tetrahedron grids. AIAA Paper 2013. 2013-0687.
[48] Batten P, Leschziner MA, Goldberg UC. Average-state Jacobians and implicit methods for compressible viscous and turbulent flows. J Comput Phys 1997;137(1):38–78.
[49] Luo H, Luo L, Nourgaliev R. A reconstructed discontinuous Galerkin method for the Euler equations on arbitrary grids. Commun Comput Phys 2012;12(5):1495–519.
[50] Luo H, Xia Y, Nourgaliev R. A class of reconstructed discontinuous Galerkin methods in computational fluid dynamics. In: International conference on mathematics and computational methods applied to nuclear science and engineering (M&C2011); 2011.
[51] Luo H, Baum JD, Löhner R. A fast, p-multigrid discontinuous Galerkin method for compressible flows at all speeds. AIAA Paper 2006;110:2006.
[52] Luo H, Baum JD, Löhner R. A *p*-multigrid discontinuous Galerkin method for the Euler equations on unstructured grids. J Comput Phys 2006;211(2):767–83.
[53] Xu Z, Liu Y, Du H, Lin G, Shu CW. Point-wise hierarchical reconstruction for discontinuous Galerkin and finite volume methods for solving conservation laws. J Comput Phys 2011;230(17):6843–65.
[54] Luo H, Baum JD, Löhner R. A fast, p-multigrid discontinuous Galerkin method for compressible flows at all speeds. AIAA J 2008;46(3):635–52.
[55] Xia Y, Luo H, Spiegel S, Frisbey M, Nourgaliev RH. A parallel, implicit reconstructed discontinuous Galerkin method for the compressible flows on 3D arbitrary grids. AIAA Paper 2013:2013–3062.
[56] Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput 1986;7(3):856–69.
[57] Cockburn B, Karniadakis G, Shu CW. The development of discontinuous Galerkin method. In: Cockburn B, Karniadakis G, Shu CW, editors. Discontinuous Galerkin methods, theory, computation, and applications. Lecture notes in computational science and engineering, vol. 11; 2000. p. 5–50.
[58] Helenbrook BT, Mavriplis D, Atkins HL. Analysis of p-multigrid for continuous and discontinuous finite element discretizations. AIAA Paper 2003:2003–3989.
[59] Luo H, Baum JD, Löhner R. On the computation of steady-state compressible flows using a discontinuous Galerkin method. Int J Numer Methods Eng 2008;73(5):597–623.
[60] Luo H, Baum JD, Löhner R. A fast, matrix-free implicit method for compressible flows on unstructured grids. J Comput Phys 1998;146(2):664–90.
[61] Luo H, Sharov D, Baum JD. On the computation of compressible turbulent flows on unstructured grids. AIAA Paper; 2000. 2000-926.
[62] Wang L, Mavriplis DJ. Implicit solution of the unsteady Euler equations for high-order accurate discontinuous Galerkin discretizations. J Comput Phys 2007;225(2):1994–2015.
[63] Erwin T, Anderson W, Kapadia S, Wang L. Three dimensional stabilized finite elements for compressible Navier–Stokes. AIAA Paper 2011;2011–3411.
[64] Hascoët L, Pascual V, et al. TAPENADE 2.1 user's guide; 2004.
[65] Karypis G, Kumar V. Metis-unstructured graph partitioning and sparse matrix ordering system, version 2.0; 1995.
[66] Schmitt V, Charpin F. Pressure distributions on the onera-m6-wing at transonic Mach numbers. Exp Data Base Comput Program Assess 1979:B1–1.
[67] Illeim ER. CFD wing/pylon/finned store mutual interference wind tunnel experiment. AEDC-TSR-91-P4. Arnold Engineering Development Center, Arnold AFB; 1991.
[68] Hummel D. Study of the flow around sharp-edged slender delta wings with large angles of attack. NASA technical translation. NASA-TT-F-15107; 1973.
[69] Thomas J, Kris S, Anderson W. Navier–Stokes computations of vortical flows over low-aspect-ratio wings. AIAA J 1990;28(2):205–12.

Computers & Fluids 96 (2014) 406–421

Contents lists available at ScienceDirect

Computers & Fluids

journal homepage: www.elsevier.com/locate/compfluid

An implicit Hermite WENO reconstruction-based discontinuous Galerkin method on tetrahedral grids

Yidong Xia [a,*], Hong Luo [a], Robert Nourgaliev [b]

[a] North Carolina State University, Raleigh, NC 27695, USA
[b] Lawrence Livermore National Laboratory, Livermore CA 94551, USA

ARTICLE INFO

Article history:
Received 26 July 2013
Accepted 20 February 2014
Available online 15 March 2014

Keywords:
Discontinuous Galerkin
Reconstruction method
WENO
Compressible flows
Implicit method
Tetrahedral grids

ABSTRACT

An Implicit Reconstructed Discontinuous Galerkin method, IRDG (P_1P_2), is presented for solving the compressible Euler equations on tetrahedral grids. In this method, a quadratic polynomial (P_2) solution is first reconstructed using a least-squares method from the underlying linear polynomial (P_1) DG solution. By taking advantage of the derivatives in the DG formulation, the stencils used in the reconstruction involve only von Neumann neighborhood (adjacent face-neighboring cells) and thus are compact and consistent with the underlying DG method. The final P_2 solution is then obtained using a WENO reconstruction, which is necessary to ensure stability of the RDG (P_1P_2) method. A matrix-free GMRES (generalized minimum residual) algorithm is presented to solve the approximate system of linear equations arising from Newton linearization. The LU-SGS (lower–upper symmetric Gauss–Seidel) preconditioner is applied with both the simplified and approximate Jacobian matrices. The numerical experiments on a variety of flow problems demonstrate that the developed IRDG (P_1P_2) method is able to obtain a speedup of at least two orders of magnitude than its explicit counterpart, maintain the linear stability, and achieve the designed third order of accuracy: one order of accuracy higher than the underlying second-order DG (P_1) method without significant increase in computing costs and storage requirements. It is also found that a well approximated Jacobian matrix is essential for the IRDG method to achieve fast converging speed and maintain robustness on large-scale problems.

1. Introduction

The discontinuous Galerkin methods [1–25] combine two advantageous features commonly associated to finite element and finite volume methods. As in classical finite element methods, accuracy is obtained by means of high-order polynomial approximation within an element rather than by wide stencils as in the case of finite volume methods. The physics of wave propagation is, however, accounted for by solving the Riemann problems that arise from the discontinuous representation of the solution at element interfaces. In this respect, the methods are therefore similar to finite volume methods. In contrast to the enormous advances in the theoretical and numerical analysis of the DG methods, the development of a viable, attractive, competitive, and ultimately superior DG method over the more mature and well-established second order methods is relatively an untouched area. This is mainly due to the fact that the DG methods have a number of weaknesses that have yet to be addressed, before they can be robustly used to flow problems of practical interest in a complex configuration environment. In particular, there are three most challenging and unresolved issues in the DG methods: (a) how to efficiently discretize diffusion terms required for the Navier–Stokes equations, (b) how to effectively control spurious oscillations in the presence of strong discontinuities, and (c) how to develop efficient time integration schemes for time accurate and steady-state solutions. Indeed, compared to the finite element methods and finite volume methods, the DG methods require solutions of systems of equations with more unknowns for the same grids. Consequently, these methods have been recognized as expensive in terms of both computational costs and storage requirements.

Dumbser et al. [18–20] have originally introduced a new family of reconstructed DG (RDG) methods, termed P_nP_m schemes, where P_n indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and P_m represents a reconstructed polynomial solution of degree of m ($m \geqslant n$) that is used to compute the fluxes. The beauty of P_nP_m schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of P_nP_m schemes, and thus allow for a direct efficiency comparison.

* Corresponding author. Tel.: +1 919 534 5645.
E-mail address: yxia2@ncsu.edu (Y. Xia).

http://dx.doi.org/10.1016/j.compfluid.2014.02.027
0045-7930/

Obviously, the construction of an accurate and efficient reconstruction operator is crucial to the success of the P_nP_m schemes. In Dumbser's work, this is achieved using a so-called in-cell recovery. The resultant over-determined system is then solved using a least-squares method that guarantees exact conservation, not only of the cell averages but also of all higher order moments in the reconstructed cell itself, such as slopes and curvatures. However, this conservative least-squares recovery approach is computationally expensive, as it involves both recovery of a polynomial solution of higher order and least-squares solution of the resulting over-determined system. Furthermore, the recovery might be problematic for a boundary cell, where the number of the face-neighboring cells might be not enough to provide the necessary information to recover a polynomial solution of a desired order.

Fortunately, recovery is not the only way to obtain a polynomial solution of higher order from the underlying discontinuous Galerkin solutions. Rather, reconstruction widely used in the FV methods provides an alternative, probably a better choice to obtain a higher-order polynomial representation. Luo et al. developed a reconstructed discontinuous Galerkin method using a Taylor basis [26–28] for the solution of the compressible Euler and Navier–Stokes equations on arbitrary grids, where a higher order polynomial solution is reconstructed by use of a strong interpolation, requiring point values and derivatives to be interpolated on the face-neighboring cells. The resulting over-determined linear system of equations is then solved in the least-squares sense. This reconstruction scheme only involves the von Neumann neighborhood, and thus is compact, simple, robust, and flexible. The reconstruction scheme guarantees exact conservation, not only of the cell averages but also of their slopes due to a judicious choice of our Taylor basis.

However, the attempt to naturally extend the RDG method to solve 3D Euler equations on tetrahedral grids is not successful. Like the second order cell-centered finite volume methods, i.e., RDG (P_0P_1), the resultant RDG (P_1P_2) methods are unstable. Although in general, RDG (P_0P_1) methods are stable in 2D and on Cartesian or structured grids in 3D, they suffer from the so-called linear instability on unstructured tetrahedral grids, when the reconstruction stencils only involve von Neumann neighborhood, i.e., adjacent face-neighboring cells [29]. Unfortunately, the RDG (P_1P_2) method exhibits the same linear instability, which can be overcome by using extended stencils. However, this is achieved at the expense of sacrificing the compactness of the underlying DG methods. Furthermore, these linear reconstruction-based DG methods will suffer from non-physical oscillations in the vicinity of strong discontinuities for the compressible Euler equations. Alternatively, ENO, WENO, and HWENO can be used to reconstruct a higher-order polynomial solution, which cannot only enhance the order of accuracy of the underlying DG method but also achieve both linear and non-linear stability. This type of hybrid HWENO + DG schemes has been developed on 1D and 2D structured grids by Balsara et al. [30], where the HWENO reconstruction is relatively simple and straightforward.

On the other hand, early efforts in the development of temporal discretization methods for RDG methods in 3D focused on explicit schemes [26–28]. Usually, explicit temporal discretizations such as multistage Runge–Kutta schemes are used to drive the solution to steady state. Acceleration techniques such as local time-stepping and implicit residual smoothing have also been combined in this context. In general, explicit schemes and their boundary conditions are easy to implement, vectorize and parallelize, and require only limited memory storage. However, for large-scale problems and especially for the solution of the Navier–Stokes equations, the rate of convergence slows down dramatically, resulting in inefficient solution techniques. In order to speed up convergence, an implicit temporal discretization technique for RDG methods is required.

In general, implicit methods require the solution of a linear system of equations arising from the linearization of a fully implicit scheme at each time step or iteration. The most widely used methods to solve a linear system on tetrahedron grids are iterative solution methods and approximate factorization methods. Significant efforts have been made to develop efficient iterative solution methods. These range from Gauss–Seidel to Krylov subspace methods that use a wide variety of preconditioners (see, e.g., Stoufflet [31], Batina [32], Venkatakrishnan et al. [33], Knight [34], Whitaker [35], Luo et al. [36], and Barth et al. [37]). The most successful and effective iterative method is to use the Krylov subspace methods [38] such as GMRES and BICGSTAB with an ILU (incomplete lower–upper) factorization preconditioner.

The objective of the effort discussed is to develop an Implicit Reconstructed Discontinuous Galerkin method, IRDG (P_1P_2), based on a Hermite WENO reconstruction using a Taylor basis [13] for the solution of the compressible Euler equations on unstructured tetrahedral grids. This HWENO-based IRDG method is designed not only to reduce the high computing costs of the DGM and improve the converging speed, but also to avoid spurious oscillations in the vicinity of strong discontinuities, thus effectively overcoming the two shortcomings of the DG methods and ensuring the stability of the RDG method. A matrix-free GMRES algorithm with an LU-SGS preconditioner is used to solve a system of linear equations, arising from an approximate linearization of an implicit temporal discretization at each time step. The developed IRDG method is used to compute a variety of flow problems on tetrahedral grids to demonstrate its accuracy, efficiency, and robustness. The remainder of this paper is organized as follows. The governing equations are listed in Section 2. The underlying RDG method is presented in Section 3. The implicit algorithm is presented in Section 4. Extensive numerical experiments are reported in Section 5. Concluding remarks are given in Section 6.

2. Governing equations

The Euler equations governing unsteady compressible inviscid flows can be expressed as

$$\frac{\partial \mathbf{U}(x,t)}{\partial t}+\frac{\partial \mathbf{F}_k(\mathbf{U}(x,t))}{\partial x_k}=0 \tag{2.1}$$

where the summation convention has been used. The conservative variable vector $\mathbf{U}$, and inviscid flux vector $\mathbf{F}$ are defined by

$$\mathbf{U}=\begin{pmatrix}\rho\\ \rho u_i\\ \rho e\end{pmatrix}\quad \mathbf{F}_j=\begin{pmatrix}\rho u_j\\ \rho u_i u_j+p\delta_{ij}\\ u_j(\rho e+p)\end{pmatrix} \tag{2.2}$$

Here ρ, p, and e denote the density, pressure, and specific total energy of the fluid, respectively, and u_i is the velocity of the flow in the coordinate direction x_i. The pressure can be computed from the equation of state

$$p=(\gamma-1)\rho\left(e-\frac{1}{2}u_ju_i\right) \tag{2.3}$$

which is valid for perfect gas, where γ is the ratio of the specific heats.

3. Reconstructed discontinuous Galerkin method

The governing Eq. (2.1) is discretized using a discontinuous Galerkin finite element formulation. To formulate the discontinuous Galerkin method, we first introduce the following weak formulation, which is obtained by multiplying the above conservation law by a test function $\mathbf{W}$, integrating over the domain Ω, and then performing an integration by parts,

$$\int_{\Omega} \frac{\partial \mathbf{U}}{\partial t}\mathbf{W}\, d\Omega + \int_{\Gamma} \mathbf{F}_k \mathbf{n}_k\, d\Gamma - \int_{\Omega} \mathbf{F}_k \frac{\partial \mathbf{W}}{\partial x_k}\, d\Omega = 0, \quad \forall \mathbf{W} \in V \tag{3.1}$$

where $\Gamma(=\partial\Omega)$ denotes the boundary of Ω, and $\mathbf{n}_j$ the unit outward normal vector to the boundary. We assume that the domain Ω is subdivided into a collection of non-overlapping tetrahedral elements Ω_e in 3D. We introduce the following broken Sobolev space V_h^p

$$V_h^p = \left\{ v_h \in [L_2(\Omega)]^m : v_h|_{\Omega_e} \in \left[V_p^m\right]\ \forall \Omega_e \in \Omega \right\} \tag{3.2}$$

which consists of discontinuous vector-values polynomial functions of degree p, and where m is the dimension of the unknown vector and

$$V_p^m = \mathrm{span}\left\{ \prod x_i^{\alpha_i} : 0 \leqslant \alpha_i \leqslant p, 0 \leqslant i \leqslant d \right\} \tag{3.3}$$

where α denotes a multi-index and d is the dimension of space. Then, we can obtain the following semi-discrete form by applying weak formulation on each element Ω_e, find $\mathbf{U}_h \in V_h^p$ such as

$$\frac{d}{dt}\int_{\Omega_e} \mathbf{U}_h \mathbf{W}_h\, d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k \mathbf{W}_h\, d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h)\frac{\partial \mathbf{W}_h}{\partial x_k}\, d\Omega = 0, \quad \forall \mathbf{W}_h \in V_h^p \tag{3.4}$$

where $\mathbf{U}_h$ and $\mathbf{W}_h$ represent the finite element approximations to the analytical solution $\mathbf{U}$ and the test function $\mathbf{W}$ respectively, and they are approximated by a piecewise polynomial function of degree p, which are discontinuous between the cell interfaces. Assume that B is the basis of polynomial function of degrees p, this is then equivalent to the following system of N equations,

$$\frac{d}{dt}\int_{\Omega_e} \mathbf{U}_h B_i\, d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k B_i\, d\Gamma - \int_{\Omega_e} \mathbf{F}_k(\mathbf{U}_h)\frac{\partial B_i}{\partial x_k}\, d\Omega = 0, \quad 1 \ll i \ll N \tag{3.5}$$

where N is the dimension of the polynomial space. Since the numerical solution $\mathbf{U}_h$ is discontinuous between element interfaces, the interface fluxes are not uniquely defined. The flux function $\mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k$ appearing in the second terms of Eq. (3.5) is replaced by a numerical Riemann flux function $\mathbf{H}_k\left(\mathbf{U}_h^L, \mathbf{U}_h^R, \mathbf{n}_k\right)$ where $\mathbf{U}_h^L$ and $\mathbf{U}_h^R$ are the conservative state vector at the left and right side of the element boundary. This scheme is called discontinuous Galerkin method of degree P, or in short notation DG (P) method. Note that the DG formulations are very similar to finite volume schemes, especially in their use of numerical fluxes. Indeed, the classical first-order cell-centered finite volume scheme exactly corresponds to the DG (P_0) method, i.e., to the DG method using a piecewise constant polynomial. Consequently, the DG (P_k) methods with $k > 0$ can be regarded as a natural generalization of finite volume methods to higher order methods. By simply increasing the degree P of the polynomials, the DG methods of corresponding higher order are obtained.

In the traditional DG method, numerical polynomial solutions $\mathbf{U}_h$ in each element are expressed using either standard Lagrange finite element or hierarchical node-based basis as following

$$\mathbf{U}_h = \sum_{i=1}^{N} \mathbf{U}_i(t) B_i(x) \tag{3.6}$$

where B_i is the finite element basis function. As a result, the unknowns to be solved are the variables at the nodes $\mathbf{U}_i$. On each cell, a system of $N \times N$ has to be solved, where polynomial solutions are dependent on the shape of elements. For example, for a linear polynomial approximation in 3D, a linear polynomial is used for tetrahedral elements and the unknowns to be solved are the variables at the four nodes. However, numerical polynomial solutions $\mathbf{U}$ can be expressed in other forms as well. In the present work, the numerical polynomial solutions are represented using a Taylor series expansion at the center of the cell. For example, if we do a Taylor series expansion at the cell centroid, the quadratic polynomial solutions can be expressed as follows

$$\begin{aligned}\mathbf{U}_h = \mathbf{U}_c &+ \left.\frac{\partial \mathbf{U}}{\partial x}\right|_c (x - x_c) + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c (y - y_c) + \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c (z - z_c) \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c \frac{(x - x_c)^2}{2} + \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c \frac{(y - y_c)^2}{2} + \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c \frac{(z - z_c)^2}{2} \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial y}\right|_c (x - x_c)(y - y_c) + \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial z}\right|_c (x - x_c)(z - z_c) \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial y \partial z}\right|_c (y - y_c)(z - z_c)\end{aligned} \tag{3.7}$$

which can be further expressed as cell-averaged values and their derivatives at the center of the cell

$$\begin{aligned}\mathbf{U}_h = \widetilde{\mathbf{U}} &+ \left.\frac{\partial \mathbf{U}}{\partial x}\right|_c B_2 + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c B_3 + \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c B_4 + \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c B_5 + \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c B_6 \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c B_7 + \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial y}\right|_c B_8 + \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial z}\right|_c B_9 + \left.\frac{\partial^2 \mathbf{U}}{\partial y \partial z}\right|_c B_{10}\end{aligned} \tag{3.8}$$

where $\widetilde{\mathbf{U}}$ is the mean value of $\mathbf{U}$ in this cell and the ten basis functions are as below. The unknowns to be solved in this formulation are the cell-averaged variables and their derivatives at the center of the cells. The dimension of the polynomial space is ten and the ten basis functions are

$$\begin{aligned}&B_1 = 1,\ B_2 = x - x_c,\ B_3 = y - y_c,\ B_4 = z - z_c \\ &B_5 = \frac{B_2^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{B_2^2}{2}\, d\Omega,\ B_6 = \frac{B_3^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{B_3^2}{2}\, d\Omega, \\ &B_7 = \frac{B_4^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e} \frac{B_4^2}{2}\, d\Omega \\ &B_8 = B_2 B_3 - \frac{1}{\Omega_e}\int_{\Omega_e} B_2 B_3\, d\Omega,\ B_9 = B_2 B_4 - \frac{1}{\Omega_e}\int_{\Omega_e} B_2 B_4\, d\Omega, \\ &B_{10} = B_3 B_4 - \frac{1}{\Omega_e}\int_{\Omega_e} B_3 B_4\, d\Omega\end{aligned} \tag{3.9}$$

The discontinuous Galerkin formulation then leads to the following ten equations

$$\frac{d}{dt}\int_{\Omega_e} \widetilde{\mathbf{U}}\, d\Omega + \int_{\Gamma_e} \mathbf{F}_k(\mathbf{U}_h)\mathbf{n}_k\, d\Gamma = 0, \quad i = 1 \tag{3.10}$$

$$\boldsymbol{M}_{9\times 9}\frac{d}{dt}\left(\left.\frac{\partial \mathbf{U}}{\partial x}\right|_c \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial y}\right|_c \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial z}\right|_c \left.\frac{\partial^2 \mathbf{U}}{\partial y \partial z}\right|_c \right)^T + \mathbf{R}_{9\times 1} = 0$$

Note that in this formulation, equations for the cell-averaged variables are decoupled from the equations for their derivatives due to the judicial choice of the basis functions and the fact that

$$\int_{\Omega_e} B_1 B_i\, d\Omega = 0, \quad 2 \leqslant i \leqslant 10 \tag{3.11}$$

In the implementation of this DG method, the basis functions are actually normalized in order to improve the conditioning of the system matrix (3.5) as follows:

$$B_1 = 1,\ B_2 = \frac{x - x_c}{\Delta x},\ B_3 = \frac{y - y_c}{\Delta y},\ B_4 = \frac{z - z_c}{\Delta z}$$

$$B_5 = \frac{B_2^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_2^2}{2}\,d\Omega,\ B_6 = \frac{B_3^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_3^2}{2}\,d\Omega,$$
$$B_7 = \frac{B_4^2}{2} - \frac{1}{\Omega_e}\int_{\Omega_e}\frac{B_4^2}{2}\,d\Omega \tag{3.12}$$

$$B_8 = B_2B_3 - \frac{1}{\Omega_e}\int_{\Omega_e}B_2B_3\,d\Omega,\ B_9 = B_2B_4 - \frac{1}{\Omega_e}\int_{\Omega_e}B_2B_4\,d\Omega,$$
$$B_{10} = B_3B_4 - \frac{1}{\Omega_e}\int_{\Omega_e}B_3B_4\,d\Omega$$

where $\Delta x = 0.5(x_{max} - x_{min})$, $\Delta y = 0.5(y_{max} - y_{mn})$, and $\Delta z = 0.5(z_{max} - z_{min})$ and x_{max}, y_{max}, z_{max} and x_{min}, y_{min}, z_{min} are the maximum and minimum coordinates in the cell Ω_e in x-, y- and z- directions, respectively. A quadratic polynomial solution can then be rewritten as

$$\begin{aligned}\mathbf{U}_h = \widetilde{\mathbf{U}} &+ \left.\frac{\partial \mathbf{U}}{\partial x}\right|_c \Delta x B_2 + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_c \Delta y B_3 + \left.\frac{\partial \mathbf{U}}{\partial z}\right|_c \Delta z B_4 + \left.\frac{\partial^2 \mathbf{U}}{\partial x^2}\right|_c \Delta x^2 B_5 \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial y^2}\right|_c \Delta y^2 B_6 + \left.\frac{\partial^2 \mathbf{U}}{\partial z^2}\right|_c \Delta z^2 B_7 + \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial y}\right|_c \Delta x \Delta y B_8 \\ &+ \left.\frac{\partial^2 \mathbf{U}}{\partial x \partial z}\right|_c \Delta x \Delta z B_9 + \left.\frac{\partial^2 \mathbf{U}}{\partial y \partial z}\right|_c \Delta y \Delta z B_{10}\end{aligned} \tag{3.13}$$

The above normalization is especially important to alleviate the stiffness of the system matrix for higher-order DG approximations.

This formulation allows us to clearly see the similarities and differences between the DG and FV methods. In fact, the discretized governing equations for the cell-averaged variables and the assumption of polynomial solutions on each cell are exactly the same for both methods. The only difference between them is the way how they obtain high-order (>1) polynomial solutions. In the finite volume methods, the polynomial solution of degrees p are reconstructed using information from the cell-averaged values of the flow variables, which can be obtained using either TVD/MUSCL or ENO/WENO reconstruction schemes. Unfortunately, the multi-dimensional MUSCL approach suffers from two shortcomings in the context of unstructured grids: (1) Uncertainty and arbitrariness in choosing the stencils and methods to compute the gradients in the case of linear reconstruction; This explains why a nominally second-order finite volume scheme is hardly able to deliver a formal solution of the second order accuracy in practice for unstructured grids. The situation becomes even more evident, severe, and profound, when a highly stretched tetrahedral grid is used in the boundary layers. Many studies, as reported by many researchers [39–41] have demonstrated that it is difficult to obtain a second-order accurate flux reconstruction on highly stretched tetrahedral grids and that for the discretization of inviscid fluxes, the classic 1D-based upwind schemes using median-dual finite volume approximation suffer from excessive numerical diffusion due to such skewing. (2) Extended stencils required for the reconstruction of higher-order (>1st) polynomial solutions. This is exactly the reason why the current finite-volume methods using the TVD/MUSCL reconstruction are not practical at higher order and have remained second-order on unstructured grids. When the ENO/WENO reconstruction schemes are used for the construction of a polynomial of degree p on unstructured grids, the dimension of the polynomial space $N = N\ (p,d)$ depends on the degree of the polynomials of the expansion p, and the number of spatial dimensions d. One must have three, six, and ten cells in 2D and four, ten, and twenty cells in 3D for the construction of a linear, quadratic, cubic Lagrange polynomial, respectively. Undoubtedly, it is an overwhelmingly challenging, if not practically impossible, task to judiciously choose a set of admissible and proper stencils that have such a large number of cells on unstructured grids especially for higher order polynomials and higher dimensions. This explains why the application of higher-order ENO/WENO methods hardly exists on unstructured grids, in spite of their tremendous success on structured grids and their superior performance over the MUSCL/TVD methods. Unlike the FV methods, where the derivatives are reconstructed using cell average values of the neighboring cells, the DG method computes the derivatives in a manner similar to the mean variables. This is compact, rigorous, and elegant mathematically in contrast with arbitrariness characterizing the reconstruction schemes with respect to how to compute the derivatives and how to choose the stencils used in the FV methods. It is our belief that this is one of the main reasons why the second order DG methods are more accurate than the FV methods using either TVD/MUSCL or ENO/WENO reconstruction schemes and are less dependent on the mesh regularity, which has been demonstrated numerically [13]. Furthermore, the higher order DG methods can be easily constructed by simply increasing the degree p of the polynomials locally, in contrast to the finite volume methods which use the extended stencils to achieve higher order of accuracy.

However, in comparison with reconstructed FV methods, the DG methods have a significant drawback in that they require more degrees of freedom, an additional domain integration, and more Gauss quadrature points for the boundary integration, and therefore more computational costs and storage requirements. On one hand, the reconstruction methods that FV methods use to achieve higher-order accuracy are relatively inexpensive but less accurate and robust. One the other hand, DG methods that can be viewed as a different way to extend a FV method to higher orders are accurate and robust but costly. It is only natural and tempting to combine the efficiency of the reconstruction methods and the accuracy of the DG methods. This idea was originally introduced by Dumbser et al. in the frame of P_nP_m scheme [18–20], where Pn indicates that a piecewise polynomial of degree of n is used to represent a DG solution, and Pm represents a reconstructed polynomial solution of degree of m ($m \geqslant n$) that is used to compute the fluxes and source terms. The beauty of P_nP_m schemes is that they provide a unified formulation for both finite volume and DG methods, and contain both classical finite volume and standard DG methods as two special cases of P_nP_m schemes, and thus allow for a direct efficiency comparison. When $n = 0$, i.e. a piecewise constant polynomial is used to represent a numerical solution, P0Pm is nothing but classical high order finite volume schemes, where a polynomial solution of degree m ($m \geqslant 1$) is reconstructed from a piecewise constant solution. When $m = n$, the reconstruction reduces to the identity operator, and P_n-P_m scheme yields a standard DG method. Clearly, an accurate and efficient reconstruction is the key ingredient in extending the underlying DG method to higher order accuracy. Although our discussion in this work is mainly focused on the linear DG method, its extension to higher order DG methods is straightforward. In the case of DG (P_1) method, a linear polynomial solution $\mathbf{U}_i$ in any cell i is

$$\mathbf{U}_i = \widetilde{\mathbf{U}}_i + \left.\frac{\partial \mathbf{U}}{\partial x}\right|_i \Delta x_i B_2 + \left.\frac{\partial \mathbf{U}}{\partial y}\right|_i \Delta y_i B_3 + \left.\frac{\partial \mathbf{U}}{\partial z}\right|_i \Delta z_i B_4 \tag{3.14}$$

Using this underlying linear polynomial DG solution in the neighboring cells, one can reconstruct a quadratic polynomial solution $\mathbf{U}_i^R$ as follows:

$$\mathbf{U}_i^R = \widetilde{\mathbf{U}}_i^R + \left.\frac{\partial \mathbf{U}^R}{\partial x}\right|_i \Delta x_i B_2 + \left.\frac{\partial \mathbf{U}^R}{\partial y}\right|_i \Delta y_i B_3 + \left.\frac{\partial \mathbf{U}^R}{\partial z}\right|_i \Delta z_i B_4 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x^2}\right|_i \Delta x_i^2 B_5 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial y^2}\right|_i \Delta y_i^2 B_6 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial z^2}\right|_i \Delta z_i^2 B_7 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial y}\right|_i \Delta x_i \Delta y_i B_8 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial z}\right|_i \Delta x_i \Delta z_i B_9 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial y \partial z}\right|_i \Delta y_i \Delta z_i B_{10} \quad (3.15)$$

In order to maintain the compactness of the DG methods, the reconstruction is required to only involve Von Neumann neighborhood, i.e., the adjacent cells that share a face with the cell i under consideration. There are ten degrees of freedom, and therefore ten unknowns to be determined. However, the first four unknowns can be trivially obtained, by requiring that the reconstruction scheme has to be conservative, a fundamental requirement, and the values of the reconstructed first derivatives are equal to the ones of the first derivatives of the underlying DG solution at the centroid i. Due to the judicious choice of Taylor basis in our DG formulation, these four degrees of freedom (cell average and slopes) simply coincide with the ones from the underlying DG solution, i.e.,

$$\widetilde{\mathbf{U}}_i^R = \widetilde{\mathbf{U}}_i \left.\frac{\partial \mathbf{U}^R}{\partial x}\right|_i = \left.\frac{\partial \mathbf{U}}{\partial x}\right|_i \left.\frac{\partial \mathbf{U}^R}{\partial y}\right|_i = \left.\frac{\partial \mathbf{U}}{\partial y}\right|_i \left.\frac{\partial \mathbf{U}^R}{\partial z}\right|_i = \left.\frac{\partial \mathbf{U}}{\partial z}\right|_i \quad (3.16)$$

The remaining six degrees of freedom can be determined by requiring that the reconstructed solution and its first derivatives are equal to the underlying DG solution and its first derivatives for all the adjacent face neighboring cells. Consider a neighboring cell j, one requires

$$\mathbf{U}_j = \widetilde{\mathbf{U}}_i^R + \left.\frac{\partial \mathbf{U}^R}{\partial x}\right|_i \Delta x_i B_2 + \left.\frac{\partial \mathbf{U}^R}{\partial y}\right|_i \Delta y_i B_3 + \left.\frac{\partial \mathbf{U}^R}{\partial z}\right|_i \Delta z_i B_4 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x^2}\right|_i \Delta x_i^2 B_5 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial y^2}\right|_i \Delta y_i^2 B_6 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial z^2}\right|_i \Delta z_i^2 B_7 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial y}\right|_i \Delta x_i \Delta y_i B_8 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial z}\right|_i \Delta x_i \Delta z_i B_9 + \left.\frac{\partial^2 \mathbf{U}^R}{\partial y \partial z}\right|_i \Delta y_i \Delta z_i B_{10}$$

$$\left.\frac{\partial \mathbf{U}}{\partial x}\right|_j = \left.\frac{\partial \mathbf{U}^R}{\partial x}\right|_i \Delta x_i \frac{1}{\Delta x_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x^2}\right|_i \Delta x_i^2 \frac{B_2}{\Delta x_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial y}\right|_i \Delta x_i \Delta y_i \frac{B_3}{\Delta x_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial z}\right|_i \Delta x_i \Delta z_i \frac{B_4}{\Delta x_i} \quad (3.17)$$

$$\left.\frac{\partial \mathbf{U}}{\partial y}\right|_j = \left.\frac{\partial \mathbf{U}^R}{\partial y}\right|_i \Delta y_i \frac{1}{\Delta y_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial y^2}\right|_i \Delta y_i^2 \frac{B_3}{\Delta y_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial y}\right|_i \Delta x_i \Delta y_i \frac{B_2}{\Delta y_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial y \partial z}\right|_i \Delta y_i \Delta z_i \frac{B_4}{\Delta y_i} \left.\frac{\partial \mathbf{U}}{\partial z}\right|_j = \left.\frac{\partial \mathbf{U}^R}{\partial z}\right|_i \Delta z_i \frac{1}{\Delta z_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial z^2}\right|_i \Delta z_i^2 \frac{B_4}{\Delta z_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial x \partial z}\right|_i \Delta x_i \Delta z_i \frac{B_2}{\Delta z_i} + \left.\frac{\partial^2 \mathbf{U}^R}{\partial y \partial z}\right|_i \Delta y_i \Delta z_i \frac{B_3}{\Delta z_i}$$

where the basis function B is evaluated at the center of cell j, i.e., $B = B(x_j, y_j)$. Its matrix form can be written as

$$\begin{pmatrix} B_5^j & B_6^j & B_7^j & B_8^j & B_9^j & B_{10}^j \\ B_2^j & 0 & 0 & B_3^j & B_4^j & 0 \\ 0 & B_3^j & 0 & B_2^j & 0 & B_4^j \\ 0 & 0 & B_4^j & 0 & B_2^j & B_3^j \end{pmatrix} \begin{pmatrix} \mathbf{U}_{xxi}^R \\ \mathbf{U}_{yyi}^R \\ \mathbf{U}_{zzi}^R \\ \mathbf{U}_{xyi}^R \\ \mathbf{U}_{xzi}^R \\ \mathbf{U}_{yzi}^R \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1^j \\ \mathbf{R}_2^j \\ \mathbf{R}_3^j \\ \mathbf{R}_4^j \end{pmatrix} \quad (3.18)$$

where $\mathbf{R}$ is used to represent the right-hand-side for simplicity. Similar equations could be written for all cells connected to the cell i with a common face, which leads to a non-square matrix. The number of face-neighboring cells for a tetrahedral and hexahedral cell is four and six, respectively. As a result, the size of the resulting non-square matrix is 16×6 and 24×6, respectively. This over-determined linear system of 16 or 24 equations for six unknowns can be solved in the least-squares sense. In the present work, it is solved using a normal equation approach, which, by pre-multiplying through by matrix transpose, yields a symmetric linear system of equations 6×6. The linear system of 6×6 can be then trivially solved to obtain the second derivatives of the reconstructed quadratic polynomial solution.

This linear reconstruction-based RDG (P_1P_2) method is able to achieve the designed third order of accuracy and significantly improve the accuracy of the underlying second-order DG method for solving the 2D compressible Euler equations on arbitrary grids [42–46]. However, when used to solve the 3D compressible Euler equations on tetrahedral grids, this RDG method suffers from the so-called linear instability, that is also observed in the second-order cell-centered finite volume methods, i.e., RDG (P_0P_1) [29]. This linear instability is attributed to the fact that the reconstruction stencils only involve the von Neumann neighborhood, i.e., adjacent face-neighboring cells [29]. The linear stability can be achieved using extended stencils, which will unfortunately sacrifice the compactness of the underlying DG methods. Also, such a linear reconstruction-based DG method will suffer from non-linear instability, leading to non-physical oscillations in the vicinity of strong discontinuities for the compressible Euler equations. Alternatively, ENO, WENO, or HWENO can be used to reconstruct a higher-order polynomial solution, which cannot only enhance the order of accuracy of the underlying DG method but also achieve both linear and non-linear stability. In the present work, the reconstructed quadratic polynomial based on the Hermite WENO on cell i is a convex combination of the least-squares reconstructed second derivatives at the cell itself and its four face-neighboring cells

$$\left.\frac{\partial^2 \mathbf{U}}{\partial x_i \partial x_j}\right|_i = \sum_{k=1}^{5} w_k \left.\frac{\partial^2 \mathbf{U}}{\partial x_i \partial x_j}\right|_k \quad (3.19)$$

where the weights w_k are computed as

$$w_k = \frac{(\varepsilon + o_k)^{-\gamma}}{\sum_{i=1}^{5} (\varepsilon + o_i)^{-\gamma}} \quad (3.20)$$

where ε is a small positive number used to avoid division by zero, o_k the oscillation indicator for the reconstructed second order polynomials, and γ an integer parameter to control how fast the non-linear weights decay for non-smooth stencils. The oscillation indicator is defined as

$$o_k = \left[\int_{\Omega_i} \left(\left.\frac{\partial^2 \mathbf{U}}{\partial x_i \partial x_j}\right|_k \right)^2 d\Omega \right]^2 \quad (3.21)$$

Note that the least-squares reconstructed polynomial at the cell itself serves as the central stencil and the least-squares reconstructed polynomials on its four face-neighboring cells act as biased stencils in this WENO reconstruction. This reconstructed quadratic polynomial solution is then used to compute the domain and boundary integrals of the underlying DG (P_1) method in Eq. (3.5). The resulting RDG (P_1P_2) method, is expected to have third order of accuracy at a moderate increase of computing costs in comparison to the underlying DG (P_1) method. The extra costs are mainly due to the least-squares reconstruction, which is relatively cheap in comparison to the evaluation of fluxes, and an extra Gauss quadrature point, which is required to calculate both domain and boundary integrals. In comparison to DG (P_2), this represents a significant saving in terms of flux evaluations. Furthermore, the number of degree of freedom is greatly reduced, which leads to a significant reduction in memory requirements, and from which implicit methods will

benefit tremendously. The cost analysis for the RDG (P_1P_1) (DG (P_1)), RDG (P_1P_2) and RDG (P_2P_2) (DG (P_2)) is summarized in Table 1, where the memory requirement for storing only the implicit diagonal matrix is given as well, and which grows quadratically with the order of the DG methods. The storage requirement for the implicit DG methods is extremely demanding, especially for higher-order DG methods.

4. Implicit time discretization

The spatially discretized governing equations must be integrated in time to obtain a steady-state solution. In the previous work [42], the Runge–Kutta time-stepping method was used to compute the time integration with a slow converging speed. In order to efficiently converge the solution, implicit time integration methods must be used for the reconstructed discontinuous Galerkin methods. Eq. (3.5) can be written in a semi-discrete form as

$$\boldsymbol{V}_i \frac{\partial \mathbf{U}_i}{\partial t} = \mathbf{R}_i \tag{4.1}$$

where $\boldsymbol{V}_i$ is the mass matrix of the cell i, and $\mathbf{R}_i$ is the right hand side residual and equals zero for a steady-state solution. Using the Euler implicit time integration, Eq. (4.1) can be further written in discrete form as

$$\boldsymbol{V}_i \frac{\Delta \mathbf{U}_i^n}{\Delta t} = \mathbf{R}_i^{n+1} \tag{4.2}$$

where Δt is the time increment and $\Delta \mathbf{U}^n$ is the difference of conservative variables between time levels n and $n+1$, i.e.,

$$\Delta \mathbf{U}^n = \mathbf{U}^{n+1} - \mathbf{U}^n \tag{4.3}$$

Eq. (4.2) can be linearized in time as

$$\boldsymbol{V}_i \frac{\Delta \mathbf{U}_i^n}{\Delta t} = \mathbf{R}_i^n + \frac{\partial \mathbf{R}_i^n}{\partial \mathbf{U}} \Delta \mathbf{U}_i \tag{4.4}$$

Writing the equation for all cells leads to the delta form of the backward Euler scheme

$$\boldsymbol{A} \Delta \mathbf{U} = \mathbf{R} \tag{4.5}$$

where

$$\boldsymbol{A} = \frac{\boldsymbol{V}}{\Delta t}\mathbf{I} - \frac{\partial \mathbf{R}^n}{\partial \mathbf{U}} \tag{4.6}$$

Note that as Δt tends to infinity, the scheme reduces to the standard Newton's method for solving a system of nonlinear equations. It's known that the Newton's method has a quadratic convergence property. The term $\partial \mathbf{R}/\partial \mathbf{U}$ represents symbolically the Jacobian matrix. It involves the linearization of both inviscid and viscous flux vectors. To obtain the quadratic convergence of Newton's method, the linearization of the numerical flux function must be virtually exact. However, explicit formation of the Jacobian matrix resulting from the exact linearization of the third order numerical flux functions using the reconstructed unknown variables for inviscid fluxes requires excessive storage and is extremely expensive. An approximated second-order representation of the numerical fluxes is linearized. In each cell, the conservative variables are evaluated at every Gauss quadrature point with all ten degrees of freedom (including the reconstructed six degrees), the same with DG (P_2), while the matrix $\boldsymbol{A}$ maintains a DG (P_1) structure ($(4 \times neqns) \times (4 \times neqns)$), where $neqns$ (=5 in 3D) is the number of the unknown variables. Although the number of time steps may increase, the memory requirement and cost per time step is significantly reduced: it takes less CPU time and fewer Gauss quadrature points than implicit DG (P_2) methods to compute the Jacobian matrix, thus reduces computational cost to solve the resulting linear system.

The following flux function of Lax–Friedrichs scheme, which was previously used to obtain the left-hand-side (LHS) Jacobian matrix for the implicit finite volume methods, has showed its high efficiency for both 2D and 3D compressible flow problems [36].

$$\mathbf{R}_i(\mathbf{U}_i, \mathbf{U}_i, \mathbf{n}_{ij}) = \frac{1}{2}(\mathbf{F}(\mathbf{U}_i, \mathbf{n}_{ij}) + \mathbf{F}(\mathbf{U}_j, \mathbf{n}_{ij})) - \frac{1}{2}|\lambda_{ij}|(\mathbf{U}_j - \mathbf{U}_i) \tag{4.7}$$

where

$$\lambda_{ij} = |\mathbf{V}_{ij} \cdot \mathbf{n}_{ij}| + C_{ij} \tag{4.8}$$

where $\mathbf{V}_{ij}$ is the velocity vector and C_{ij} is the speed of sound. The linearization of the flux function yields

$$\frac{\partial \mathbf{R}_i}{\partial \mathbf{U}_i} = \frac{1}{2}(J(\mathbf{U}_i) + |\lambda_{ij}|\mathbf{I}) \tag{4.9}$$

$$\frac{\partial \mathbf{R}_i}{\partial \mathbf{U}_j} = \frac{1}{2}(J(\mathbf{U}_j) - |\lambda_{ij}|\mathbf{I}) \tag{4.10}$$

where $J = \partial \mathbf{F}/\partial \mathbf{U}$ represents the Jacobian of the inviscid flux vector. However, the extension of this simplified flux linearization to either the implicit DG (P_1) or implicit RDG (P_1P_2) methods on tetrahedral grids suffers from *CFL* constraints and leads to inefficient converging speed. Therefore the performance of implicit time integration would be greatly impaired if this version of Jacobian representation is applied.

Alternatively, $\partial \mathbf{F}/\partial \mathbf{U}$ can be formulated according to the method proposed by Batten et.al [47], in which the average-state Jacobians with frozen acoustic wave-speed are considered. This method can guarantee an uncompromised converging speed and the robustness is preserved. The implicit form of the HLLC flux is then given by

$$\mathbf{F}_{HLLC}^{n+1} = \begin{cases} \mathbf{F}_i^n + \frac{\partial \mathbf{F}_i}{\partial \mathbf{U}_i}\Delta \mathbf{U}_i & \text{if } S_i^n > 0 \\ (\mathbf{F}_i^*)^n + \frac{\partial \mathbf{F}_i^*}{\partial \mathbf{U}_i}\Delta \mathbf{U}_i + \frac{\partial \mathbf{F}_i^*}{\partial \mathbf{U}_j}\Delta \mathbf{U}_j & \text{if } S_i^n \leqslant 0 < S_M^n \\ (\mathbf{F}_j^*)^n + \frac{\partial \mathbf{F}_j^*}{\partial \mathbf{U}_i}\Delta \mathbf{U}_i + \frac{\partial \mathbf{F}_j^*}{\partial \mathbf{U}_j}\Delta \mathbf{U}_j & \text{if } S_M^n \leqslant 0 < S_j^n \\ \mathbf{F}_j^n + \frac{\partial \mathbf{F}_j}{\partial \mathbf{U}_j}\Delta \mathbf{U}_j & \text{if } S_j^n < 0 \end{cases} \tag{4.11}$$

where

$$S_M = \frac{\rho_j q_j (S_j - q_j) - \rho_i q_i (S_i - q_i) + p_i - p_j}{\rho_j (S_j - q_j) - \rho_i (S_i - q_i)} \tag{4.12}$$

where $q_i = \mathbf{V}_i \cdot \mathbf{n}_{ij}$ and $q_j = \mathbf{V}_j \cdot \mathbf{n}_{ij}$ and

$$S_i = \min[\lambda_1(\mathbf{U}_i), \lambda_1(U^{Roe})] \tag{4.13}$$

$$S_j = \max[\lambda_m(U^{Roe}), \lambda_m(\mathbf{U}_j)] \tag{4.14}$$

Table 1
Cost analysis for different RDG (P_mP_n) methods on tetrahedral grids.

	RDG (P_1P_1)	RDG (P_1P_2)	RDG (P_2P_2)
Number of quadrature points for boundary integrals	3	4	7
Number of quadrature points for domain integrals	4	5	11
Reconstruction	No	Yes	No
Order of accuracy	$O(h^2)$	$O(h^3)$	$O(h^3)$
Storage for implicit diagonal matrix	400 word per element	400	2500

with $\lambda_1(U^{Roe})$ and $\lambda_m(U^{Roe})$ being the smallest and largest eigenvalues of the Roe matrix [48]. Details of $\partial \mathbf{F}_i^*/\partial \mathbf{U}_i$ and $\partial \mathbf{F}_i^*/\partial \mathbf{U}_j$ can be found in Ref. [47]. By doing so, the quadratic convergence of the Newton's method can no longer be achieved because of the inexact representation of LHS in Eq. (4.5).

In this study, the linear system at each time step is solved approximately by using a preconditioned matrix-free GMRES solver. Instead of calculating and storing the full matrix, the matrix-free GMRES algorithm only requires the result of matrix–vector products, which can be approximated as

$$\frac{\partial \mathbf{R}(\mathbf{U})}{\partial \mathbf{U}}\Delta\mathbf{U} = \frac{\mathbf{R}(\mathbf{U}+\varepsilon\cdot\Delta\mathbf{U}) - \mathbf{R}(\mathbf{U})}{\varepsilon} \tag{4.15}$$

where $\mathbf{R}(\mathbf{U}+\varepsilon\cdot\Delta\mathbf{U})$ is the residual for the governing equations computed using perturbed state quantities and ε is a scalar. The parameter ε can be computed as proposed by Pernice and Walker [49]

$$\varepsilon = \in \frac{\sqrt{1+\|\mathbf{U}\|_{L^2}}}{\|\Delta\mathbf{U}\|_{L^2}} \tag{4.16}$$

where the parameter $\in$ is a user-defined input and the choice of this parameter is between round-off and truncation error. In this study $\varepsilon = 10^{-7}$ works without problem for all the computations. The GMRES process requires one flux evaluation per time step and one flux evaluation per inner GMRES iteration.

Since the speed of convergence of an iterative algorithm for a linear system depends on the condition number of the matrix $\boldsymbol{A}$, preconditioning is used to alter the spectrum and therefore accelerate the convergence speed of iterative algorithm. The preconditioning technique involves solving an equivalent preconditioned linear system,

$$\widetilde{\mathbf{A}}\Delta\widetilde{\mathbf{U}} = \widetilde{\mathbf{R}} \tag{4.17}$$

instead of the original system in Eq. (4.5), hoping that $\widetilde{\boldsymbol{A}}$ is well conditioned. Left preconditioning involves pre-multiplying the linear system with the inverse of a preconditioning matrix $\boldsymbol{P}$

$$\boldsymbol{P}^{-1}\boldsymbol{A}\Delta\mathbf{U} = \boldsymbol{P}^{-1}\mathbf{R} \tag{4.18}$$

The best preconditioning matrix for $\boldsymbol{A}$ would cluster as many eigenvalues as possible at unity. Obviously, the optimal choice of $\boldsymbol{P}$ is $\boldsymbol{A}$. Preconditioning will be cost-effective only if the additional computational work incurred for each sub-iteration, is compensated for by a reduction in the total number of iterations to convergence. In this way, the total cost of solving the overall nonlinear system is reduced.

Using an edge-based data structure, the matrix $\boldsymbol{A}$ is stored in upper, lower, and diagonal forms, which can be expressed as

$$\boldsymbol{U} = \frac{\partial \mathbf{R}_i(\mathbf{U}_i,\mathbf{U}_j,\boldsymbol{n}_{ij})}{\partial \mathbf{U}_j} \quad \boldsymbol{L} = -\frac{\partial \mathbf{R}_i(\mathbf{U}_i,\mathbf{U}_j,\boldsymbol{n}_{ij})}{\partial \mathbf{U}_i}$$
$$\boldsymbol{D} = \frac{\boldsymbol{V}}{\Delta t}\mathbf{I} + \sum_j \frac{\partial \mathbf{R}_i(\mathbf{U}_i,\mathbf{U}_j,\boldsymbol{n}_{ij})}{\partial \mathbf{U}_i} \tag{4.19}$$

where $\boldsymbol{U}$, $\boldsymbol{L}$, and $\boldsymbol{D}$ represent the strict upper matrix, the strict lower matrix, and the diagonal matrix, respectively. Both upper and lower matrices require a storage of $nedge \times (ndegr \times neqns)^2$ and the diagonal matrix needs a storage of $nelem \times (ndegr \times neqns)^2$, where $nedge$ is the number of faces, $ndegr$(=4 for RDG (P_1P_2) in 3D) is the number of degree of freedom, and $nelem$ is the number of cells. Therefore the current scheme for the preconditioner is not completely matrix free. In the present work, the LU-SGS method is used as a preconditioner [29], i.e.,

$$\boldsymbol{P} = (\boldsymbol{D}+\boldsymbol{L})\boldsymbol{D}^{-1}(\boldsymbol{D}+\boldsymbol{U}) \tag{4.20}$$

The structure of the restarted preconditioned matrix-free GMRES algorithm is described below to solve the linear system of equations at each time step.

For $l = 1,\ m$ Do	m Restarted iterations
$\mathbf{v}_0$ = $\mathbf{R} - \boldsymbol{A}\Delta\mathbf{U}_0$	Initial residual
$\mathbf{r}_0$:= $\boldsymbol{P}^{-1}\mathbf{v}_0$	Preconditioning step
$\beta := \|\mathbf{r}_0\|_2$	Initial residual norm
$\mathbf{v}_1$:= $\mathbf{r}_0/\beta$	Define initial Krylov
For $j = 1,\ k$ Do	Inner iterations
$\mathbf{y}_j := (\mathbf{R}(\mathbf{U}+\varepsilon\cdot\Delta\mathbf{U})$-$\mathbf{R}(\mathbf{U}))/\varepsilon$	Matrix–vector product
$\mathbf{w}_j = \boldsymbol{P}^{-1}\mathbf{y}_j$	Preconditioning step
For $i = 1,\ j$ Do	Gram–Schmidt step
$h_{i,j}$:= $(\mathbf{w}_j, \mathbf{v}_i)$	...
$\mathbf{w}_j = \mathbf{w}_j - h_{i,j}\mathbf{v}_i$	...
End Do	
$h_{j+1,j} := \|\mathbf{w}_j\|_2$	...
$\mathbf{v}_{j+1}$:= $\mathbf{w}_j/\mathbf{h}_{j+1,j}$	Define Krylov vector
End Do	
$\mathbf{z} := \min_z \|\beta\mathbf{e}_1 - H\hat{\mathbf{z}}\|_2$	Solve least squares
$\Delta\mathbf{U} := \mathbf{U}_0 + \sum_{i=1}^{m}\mathbf{v}_i\mathbf{z}_i$	Approximate solution
if $\|\beta\mathbf{e}_1 - H\hat{\mathbf{z}}\|_2 \ll \Delta\epsilon$ exit	Convergence check
$\Delta\mathbf{U}_0 := \Delta\mathbf{U}$	Restart
End Do	

The primary storage is dictated by the LU-SGS preconditioner, which requires the upper, lower and diagonal matrix to be stored for every non-zero element in the global matrix $\widetilde{\mathbf{A}}$ on the left hand side of Eq. (4.17).When evaluating the matrix–vector product by Eq. (4.15), the linearization of the fluxes requires a storage of $nelem \times ndegr \times neqns$. The need for additional storage associated with the GMRES algorithm is an array of size $(k+2) \times nelem \times (ndegr \times neqns)$, where k is the number of search directions. Since the preconditioned matrix-free GMRES algorithm is completely separated from the flux computation procedure, the memory which is used to compute the fluxes can also be shared in the procedure of solving the linear system.

5. Computational results

The developed implicit Hermite WENO reconstruction-based discontinuous Galerkin method has been used to solve a variety of the compressible flow problems on tetrahedral grids. All of the computations are performed on a Dell Precision T7400 personal workstation computer (2.98 GHz Xeon CPU with 18 GBytes memory) using Red Hat 5 Linux operating system. The developed implicit method is used to compete with the explicit three-stage Runge–Kutta time-stepping method for all test cases in order to demonstrate its superior efficiency. The average-state HLLC full Jacobians (Full LHS) and diagonal Jacobians (Diagonal LHS), and Lax–Friedrichs Jacobians (Simplified LHS) are used respectively as the preconditioning matrix for the implicit method in order to verify the effect of accuracy of approximate Jacobian matrix on converging speed. All computations are initiated with uniform flows and are carried out using a *CFL* number of 1 for the explicit method, and 10 for the implicit method for the first 20 steps. The setting of *CFL* number is not considered an emphasis in this paper. A few examples are presented in this section to demonstrate that the developed IRDG (P_1P_2) method is able to maintain the linear stability, achieve the designed third order of accuracy, and significantly improve the accuracy of the underlying second-order DG (P_1) method without significant increase in computing costs and storage requirements.

5.1. A subsonic flow through a channel with a smooth bump

This test case is chosen to demonstrate the convergence accuracy and efficiency of the IRDG (P_1P_2) methods for internal flows. The problem under consideration is a subsonic flow inside a 3D channel with a smooth bump on the lower surface. The height, width, and length of the channel are 0.8, 0.8, and 3, respectively. The shape of the lower wall is defined by the function $0.0625exp(-25x^2)$ from $x = -1.5$ to $x = 1.5$. The inflow condition is prescribed at a Mach number of 0.5, and an angle of attack of 0°. Fig. 1 shows the four successively refined tetrahedral grids used for the grid convergence study. The numbers of elements, points, and boundary points for the coarse, medium, fine, and finest grids are (889, 254, 171), (6986, 1555, 691), (55,703, 10,822, 2711), and (449,522, 81,567, 10,999), respectively. The cell size is halved between consecutive grids. Numerical solutions to this problem are computed using the IRDG (P_1P_1) and IRDG (P_1P_2) methods on the first three grids to obtain a quantitative measurement of the order of accuracy and discretization errors. The following L^2-norm of the entropy production is used as the error measurement

$$\|\varepsilon\|_{L_2(\Omega)} = \sqrt{\int_\Omega \varepsilon^2 d\Omega}$$

where the entropy production ε is defined as

$$\varepsilon = \frac{S - S_\infty}{S_\infty} = \frac{p}{p_\infty}\left(\frac{\rho_\infty}{\rho}\right)^\gamma - 1$$

Note that the entropy production, where the entropy is defined as $S = p/\rho^\gamma$, is a very good criterion to measure accuracy of the numerical solutions, since the flow under consideration is isentropic. Fig. 2 illustrates the computed velocity contours in the flow field obtained by the RDG (P_1P_1) and RDG (P_1P_2) methods on the fine grid. Table 2 provides the details of the spatial convergence of the RDG methods for this numerical experiment. Consider the fact that this is a 3D simulation of a 2D problem, and unstructured tetrahedral grids are not symmetric by nature, thus causing error in the z-direction, the second-order RDG (P_1P_1) method can be considered to deliver the designed second order of accuracy. As expected, the RDG (P_1P_2) method offers a full $O(h^{p+2})$ order of the convergence, adding one order of accuracy to the underlying IRDG (P_1P_1) method, thus demonstrate that the IRDG (P_1P_2) method can significantly increase the accuracy of the underlying DG method, and therefore greatly decrease its computing costs.

The *CFL* number for the implicit method is set to be 500 for all grids after the initial time steps. Figs. 3–5 shows a series of plots of logarithmic density residual versus time steps (left) and CPU time (right) for the explicit and implicit methods using RDG (P_1P_2) on different grids. In comparison, even the implicit method with only the diagonal part of Jacobians is three orders of magnitude faster than its explicit counterpart, indicating the advantages of using implicit time integration scheme under the context of the high-order reconstructed discontinuous Galerkin methods.

On the other side, comparison among the different approximated preconditioning matrix verifies that the converging speed of solution is remarkably affected by how the Jacobians are formulated. On the fine grid, the HLLC full Jacobian matrix leads in the best converging speed in terms CPU time, whereas the HLLC diagonal Jacobian matrix is about six times slower, and simplified full Jacobian matrix is two times slower. It is also found that only the

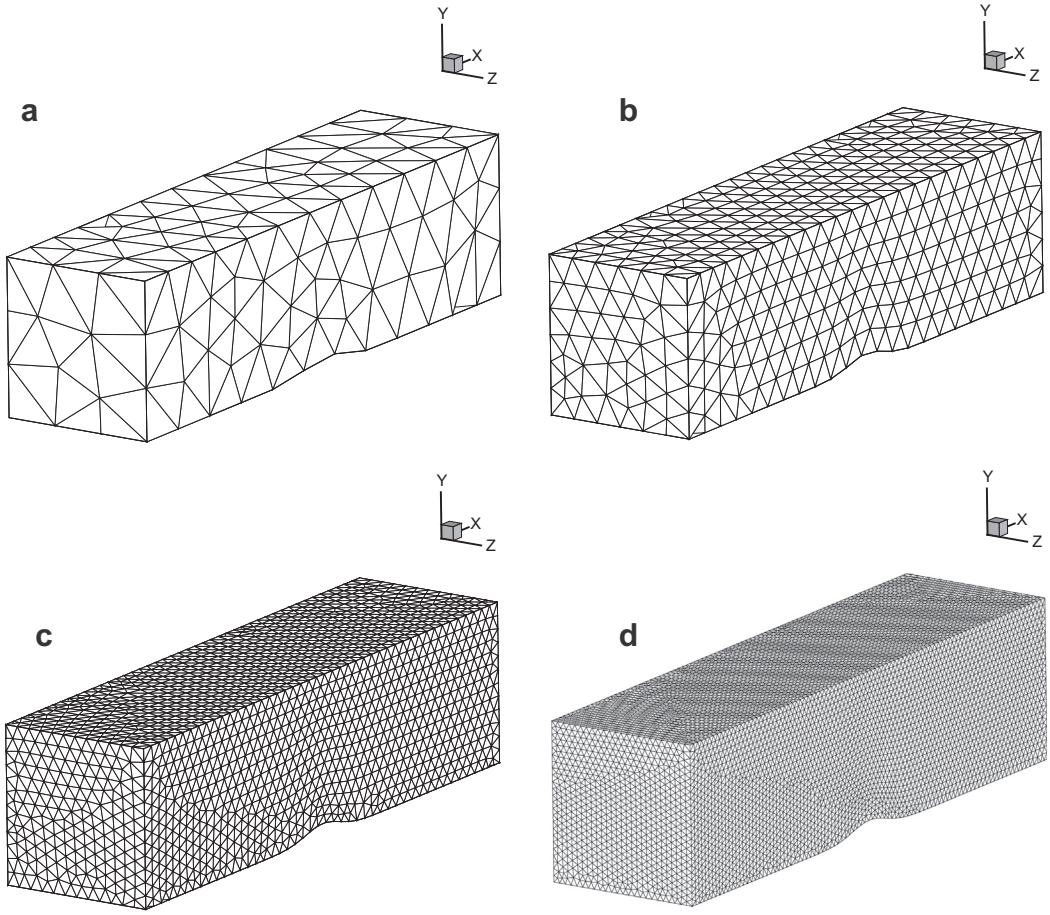

Fig. 1. A sequence of four successively globally refined unstructured meshes used for computing subsonic flow in a channel with a smooth bump.

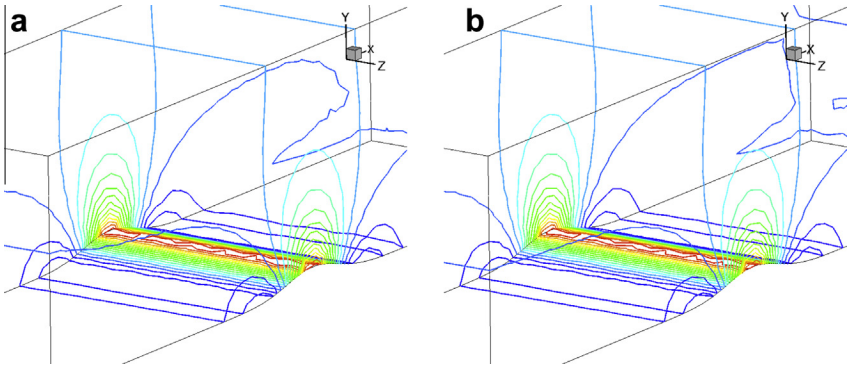

Fig. 2. Computed velocity contours in the flow field obtained by the IRDG (P_1P_1) (left) and IRDG (P_1P_2) (right) on the fine mesh for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$.

Table 2
Convergence order of accuracy for RDG (P_1P_1) and RDG (P_1P_2) methods for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$.

Cell size	Log (L^2-error)		Order	
	IRDG (P_1P_1)	IRDG (P_1P_2)	IRDG (P_1P_1)	IRDG (P_1P_2)
8X	−2.61293	−2.67090		
4X	−3.13334	−3.56436	1.89	2.80
2X	−3.74301	−4.35115		

solver with the HLLC full Jacobian matrix maintains stable convergence with *CFL* numbers like 10^4 or even larger, although the speedup does not increase obviously. In contrast, the solver with either of the other two Jacobians suffers instability when *CFL* number of 10^3. In other words, less approximated Jacobians lead to larger condition number, thus lead to more rigorous *CFL* constraints. Also see Fig. 6, the solver with HLLC full Jacobians is over three times faster than that with simplified full Jacobians on the finest grid, indicating the necessity of using well approximated Jacobians on highly refined grids. In fact, for the traditional finite volume methods in 2D and 3D, it's common practice to apply simplified Jacobians for implicit time integration without difficulty. In contrast, for high-order discontinuous Galerkin methods on tetrahedral grids, the efficiency of an implicit solver is closely linked to how accurately the numerical flux function is linearized.

5.2. A subsonic flow past a sphere

In this test case, a subsonic flow past a sphere at a Mach number of $M_\infty = 0.5$ is chosen to assess if the developed IRDG method can

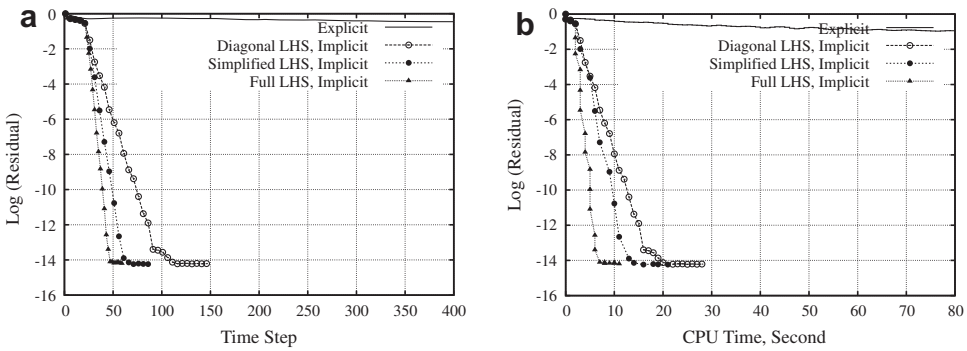

Fig. 3. Logarithmic density residual versus time step (left) and CPU time (right) for RDG (P_1P_2) on the coarse mesh for subsonic flow through a channel with a bump on the lower surface $M_\infty = 0.5$.

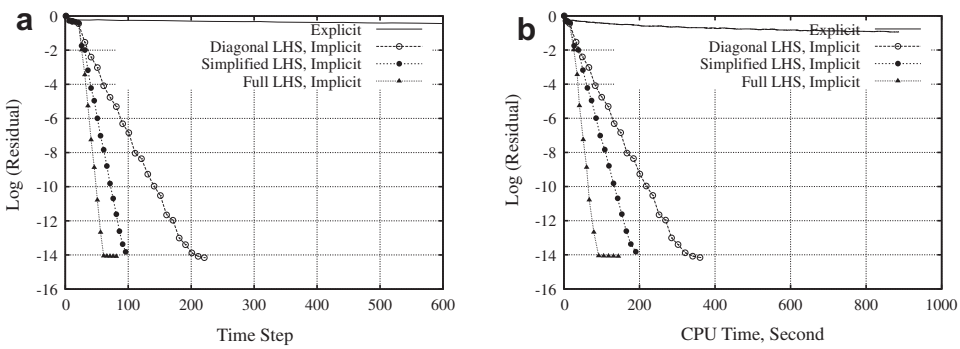

Fig. 4. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) on the medium mesh for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$.

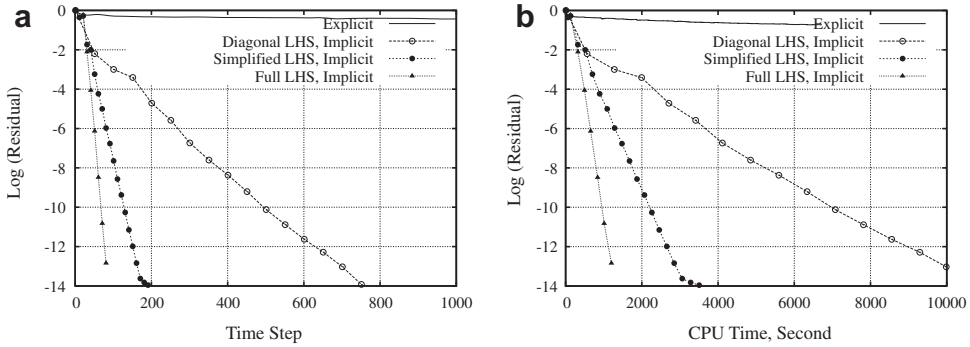

Fig. 5. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) on the fine mesh for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$.

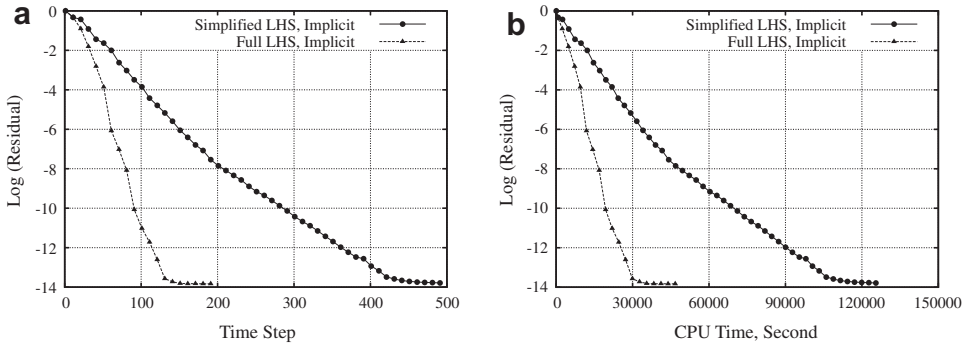

Fig. 6. Logarithmic versus time steps (left) and CPU time (right) for RDG (P_1P_2) on the finest mesh for a subsonic flow through a channel with a bump on the lower surface at $M_\infty = 0.5$.

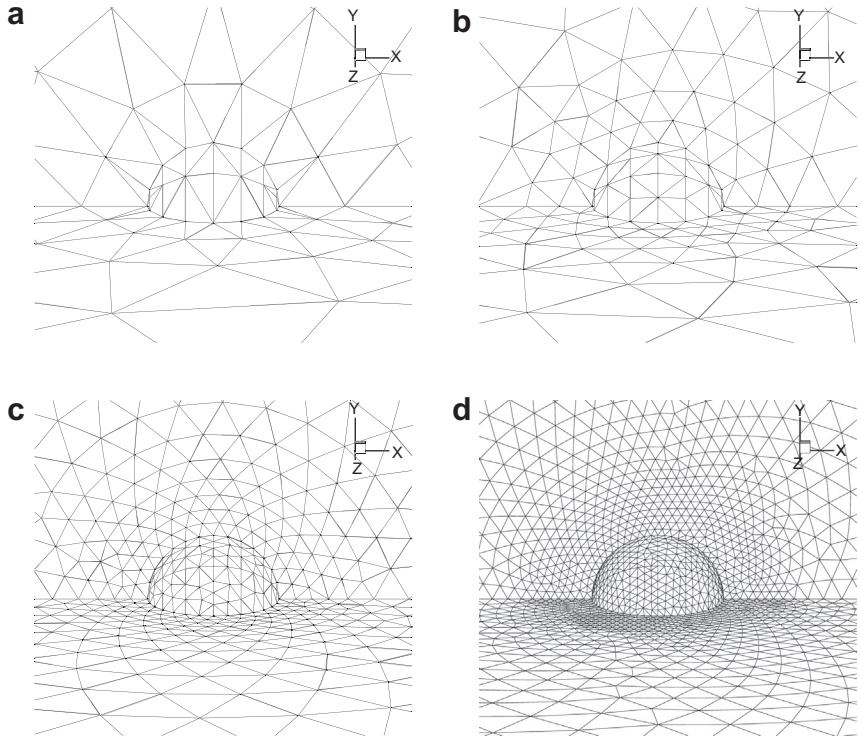

Fig. 7. A series of four successively globally refined tetrahedral meshes for computing subsonic flow past a sphere at $M_\infty = 0.5$.

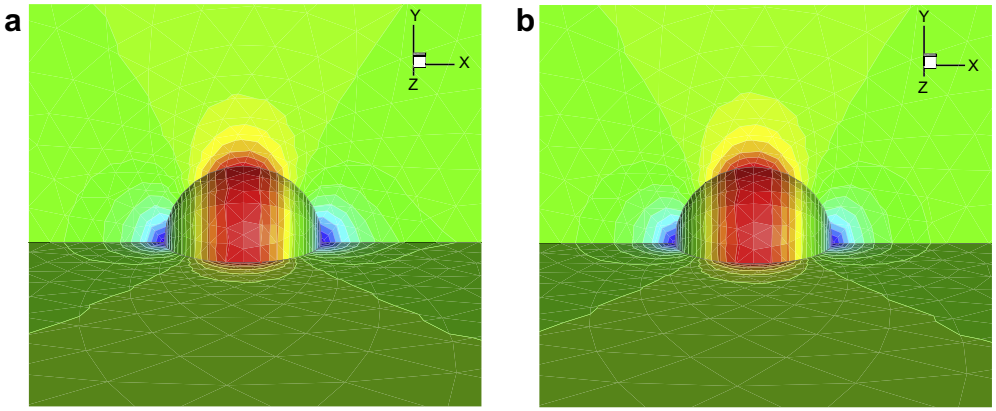

Fig. 8. Computed velocity contours in the flow field obtained by the IRDG (P_1P_1) method on the fine grid (left), and IRDG (P_1P_2) on the fine grid (right) for subsonic flow past a sphere at $M_\infty = 0.5$.

Table 3
Orders of convergence rate for RDG (P_1P_1) and RDG (P_1P_2) methods for subsonic flow past a sphere at $M_\infty = 0.5$.

Cell size	Log (L^2-error)		Mean order	
	IRDG (P_1P_1)	IRDG (P_1P_2)	IRDG (P_1P_1)	IRDG (P_1P_2)
8X	−1.74887	−1.97797		
4X	−2.30018	−2.88048	2.36	3.55
2X	−2.90949	−3.70373		

achieve a formal order of convergence rate and high converging speed for external flows. A sequence of the four successively refined tetrahedral grids used in this grid convergence study is shown in Fig. 7. The numbers of elements, points, and boundary points for the coarse, medium, fine, and finest grids are (535, 167, 124), (2426, 598, 322), (16,467, 3425, 1188), and (124,706, 23,462, 4538), respectively. The cell size is halved between consecutive meshes. Note that only a quarter of the configuration is modeled due to the symmetry of the problem, and that the number of elements on a successively refined mesh is not exactly eight times the coarse mesh's elements due to the nature of unstructured grid generation.

The computations are conducted on the first three grids using the IRDG (P_1P_1) and IRDG (P_1P_2) methods to obtain a quantitative measurement of the order of accuracy and discretization errors. As in the previous case, the entropy production is used as the error measurement. Fig. 8 illustrates the computed velocity contours in

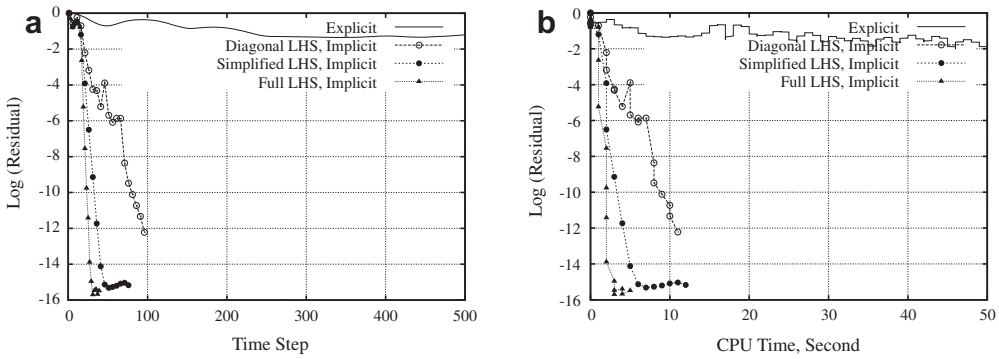

Fig. 9. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) on the coarse mesh for subsonic flow past a sphere $M_\infty = 0.5$.

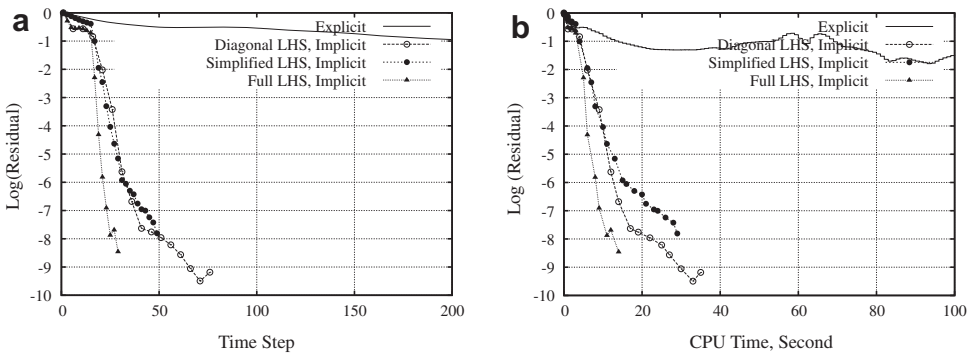

Fig. 10. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) on the medium mesh for subsonic flow past a sphere M_∞=0.5.

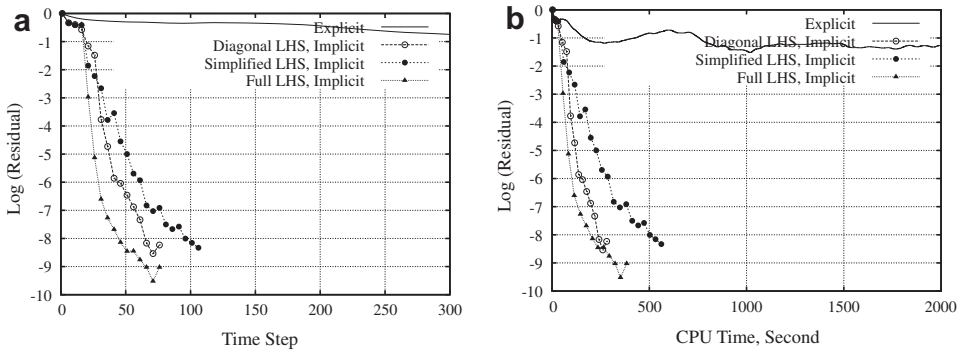

Fig. 11. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) on the fine mesh for subsonic flow past a sphere $M_\infty = 0.5$.

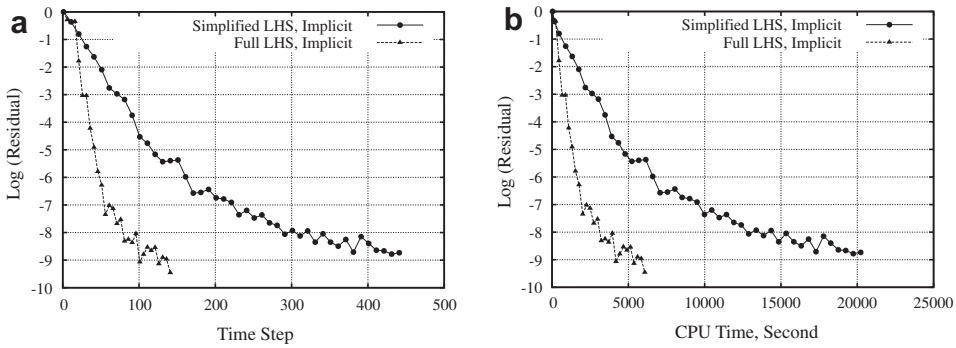

Fig. 12. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) on the finest mesh for subsonic flow past a sphere $M_\infty = 0.5$.

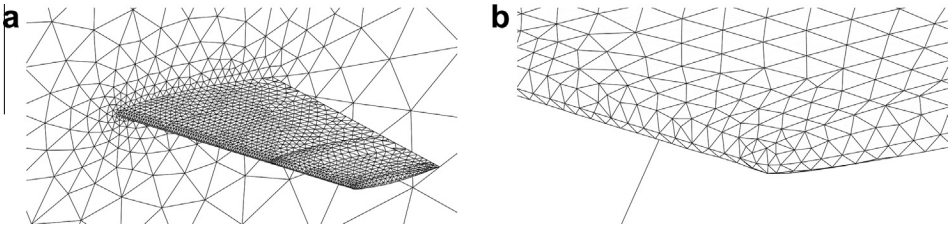

Fig. 13. Left: unstructured mesh used for computing a transonic flow past an ONERA M6 wing (41,440 elements, 8325 points, 2575 boundary points). Right: Details around the leading edge.

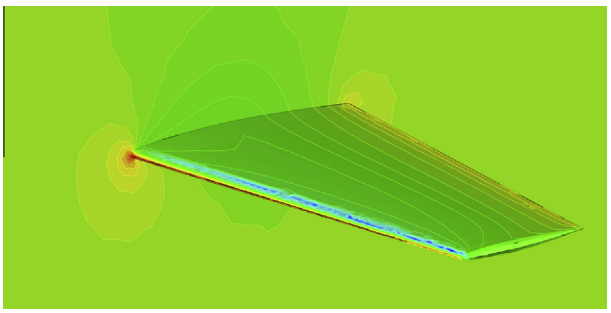

Fig. 14. Computed pressure contours obtained by the RDG (P_1P_2) method for transonic flow past an ONERA M6 wing at $M_\infty = 0.699$, $\alpha = 3.06^\circ$.

the flow field obtained by the RDG (P_1P_1) and RDG (P_1P_2) methods on the fine grid. One can observe that the RDG (P_1P_2) solution is much more accurate than the RDG (P_1P_1) solution on the same fine grid. Table. 3 provides the details of the spatial convergence of the two IRDG methods for this numerical experiment. Both the IRDG (P_1P_1) and IRDG (P_1P_2) methods have achieved higher than expected convergence rates, being 2.36 and 3.55 respectively, convincingly demonstrating the benefits of using a higher-order method. The *CFL* number is set to be 10^3 for all grids after the initial time steps. Figs. 9–11 shows a series of plots of logarithmic density residual versus time step (left) and CPU time (right) for the explicit and implicit RDG (P_1P_2) methods on the four grids. The implicit method is at least four orders of magnitude faster than its explicit counterpart in this test case, again indicating the superior advantages of using implicit time integration schemes. Among the three preconditioners, The HLLC full Jacobians still provide the best converging speed. However, the HLLC diagonal Jacobians are two times faster than the simplified full Jacobians on the fine grid, which is different than in the first test case. Again in this case, it is found that if a very large *CFL* number like 10^4 is used, only the solver with the HLLC full Jacobians is able to maintain stability, whereas with the other two Jacobians, the solver needs more initial time steps before a larger *CFL* number can be safely applied. See Fig. 12, the implicit solver with HLLC Jacobians is three times faster than that with a simplified form on the finest grid, indicating the better robustness and performance of using more accurately approximated Jacobians on highly refined grids.

5.3. Transonic flow over the ONERA M6 wing

A transonic flow over the ONERA M6 wing at a Mach number of $M_\infty = 0.699$ and an attack angle of $\alpha = 3.06^\circ$ is considered in this example. This case is chosen to demonstrate that the IRDG (P_1P_2) method is able to maintain the robustness of the underlying DG methods at the presence of weak discontinuities. The DG method is not only linear stable but also has the ability to obtain a stable solution for weak discontinuities in spite of the over- and under-shots in the vicinity of shock waves.

The mesh used in this computation consists of 41,440 elements, 8325 grid points, and 2575 boundary points, as shown in Fig. 13. One can observe the coarseness of grids even in the vicinity of the leading edge.The computed pressure contours obtained by the RDG (P_1P_2) solution on the wing surface are shown in Fig. 14. Fig. 15 compares the pressure coefficient distributions at six span-wise locations on the wing surface between the numerical results and the experimental data. The pressure coefficients are computed at the two nodes of each triangle that intersect with the cut plane, and plotted by a straight line. This representation truly reflects the discontinuous nature of the DG solution. Since no limiters and Hermite WENO-reconstruction are used to eliminate the spurious oscillations for the underlying DG (P_1) method, the over- and under-shoots in the vicinity of the shock waves are clearly visible. Besides, the calculations show a good agreement with experiment data. This example clearly demonstrates that the IRDG (P_1P_2) is able to maintain the linear stability of the underlying DG method.

The plots of logarithmic density residual versus time steps and CPU time for RDG (P_1P_2) methods are shown in Fig. 16. The implicit solver with HLLC full Jacobians at a *CFL* number of 10^5, is over 30 times faster than its explicit counterpart in terms of CPU time. It is found when the simplified Jacobians are used, stability will not be maintained if a *CFL* number of 10^5 is applied after the same initial steps. As a result, it needs 2.5 times more time steps and 3.5 times more CPU time for full convergence at a *CFL* number of 10^4 than the implicit solver with HLLC Jacobians, due to the fact that more inner iterations in the GMRES process are required for each time step.

5.4. Subsonic flow over the wing/pylon/finned-store configuration

The fourth case is for the subsonic flow over the wing/pylon/finned-store configuration reported in Ref. [50] at $M_\infty = 0.5$ and

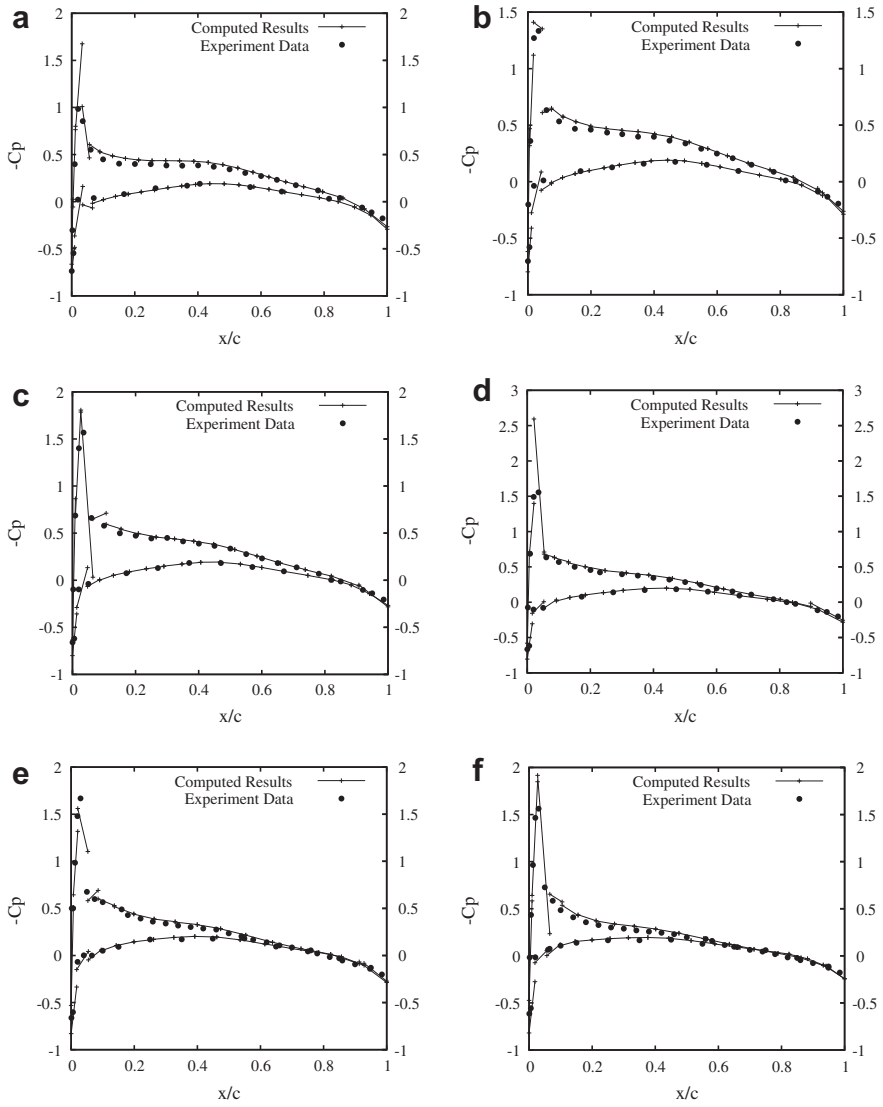

Fig. 15. Pressure coefficient distributions for ONERA M6 wing at six span-wise locations, $M_\infty = 0.699$ $\alpha = 3.06^\circ$.

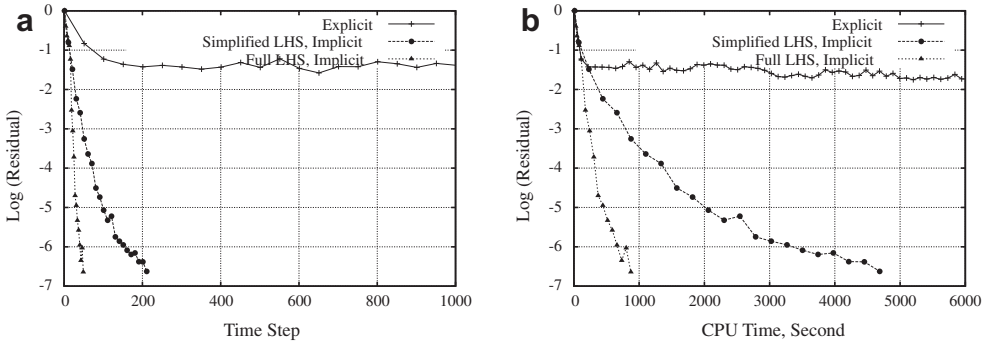

Fig. 16. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) for transonic flow past ONERA M6 wing at $M_\infty = 0.699,\ \alpha = 3.06^\circ$.

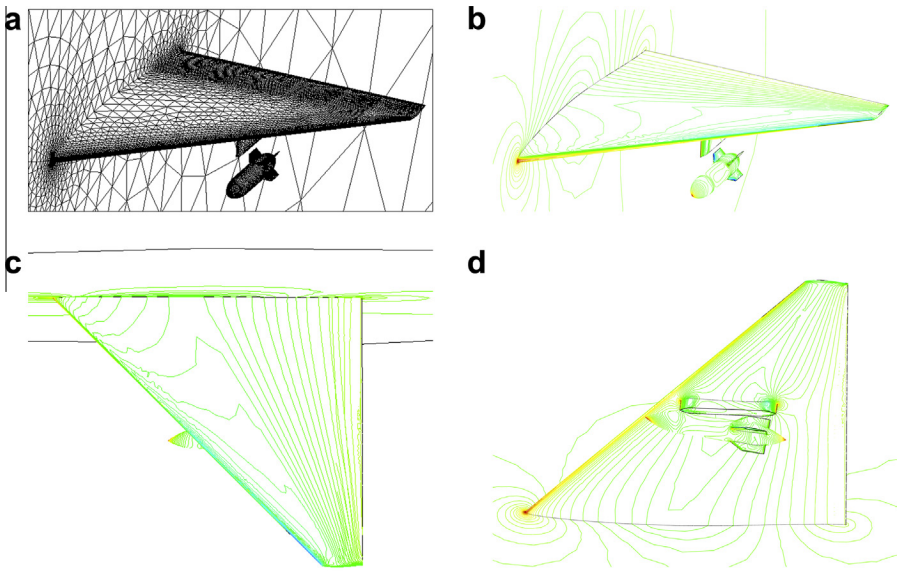

Fig. 17. (a) Surface mesh for the wing/pylon/store (328,370 elements, 62,630 points, 14,325 boundary points). (b) Computed pressure contours from the front side view at $M_\infty = 0.5,\ \alpha = 3.06^\circ$. (c) Computed pressure contours on upper surface at $M_\infty = 0.5,\ \alpha = 3.06^\circ$. (d) Computed pressure contours on lower surface at $M_\infty = 0.5,\ \alpha = 3.06^\circ$.

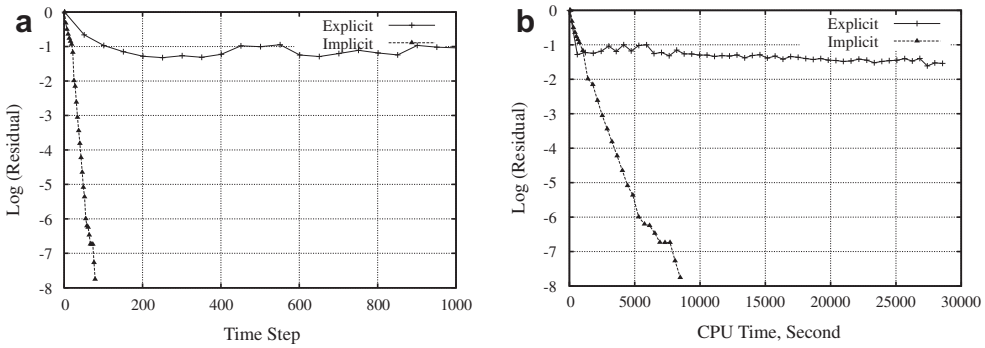

Fig. 18. Logarithmic density residual versus time steps (left) and CPU time (right) for RDG (P_1P_2) for subsonic flow past wing/pylon/store at $M_\infty = 0.5,\ \alpha = 3.06^\circ$.

$\alpha = 3.06^\circ$. This test case is conducted to test the performance of the IRDG (P_1P_2) method for computing flows over complex geometric configurations. The configuration consists of a clipped delta wing with a 45° sweep comprised from a constant NACA 64,010 symmetric airfoil section. The wing has a root chord of 15 in., a semi-span of 13 in., and a taper ratio of 0.134. The pylon is located at the midspan station and has a cross-section characterized by a flat plate closed at the leading and trailing edges by a symmetrical ogive shape. The width of the pylon is 0.294 in. The four fins on the store are defined by a constant NACA 0008 airfoil section with a leading-edge sweep of 45° and a truncated tip. The mesh used in the computation is shown in Fig.17(a). It contains 328,370 elements, 62,630 grid points, and 14,325 boundary points. Fig. 17(b)–(d) show the computed pressure contours from the front side view, on the upper and lower wing surface, respectively. Fig. 18 shows a comparison for logarithmic density residual versus

time steps and CPU time between the explicit and implicit solvers for RDG (P_1P_2). Only the implicit solver with HLLC full Jacobians is tested in this case. The IRDG (P_1P_2) solver obtains a speedup of more than two orders of magnitude than the explicit solver. In fact, the explicit solver is already not a practical choice in this case. Furthermore, it is even impossible for explicit solvers to fully converge the flow at all for some more complex configurations. Therefore the developed IRDG (P_1P_2) method has again proved its superior performance and offers the feasibility in large-scale engineering applications.

6. Conclusions

An implicit Hermite WENO reconstruction-based discontinuous Galerkin method, IRDG (P_1P_2), has been presented to solve the compressible Euler equations on tetrahedral grids. An LU-SGS preconditioned matrix-free GMRES algorithm has been applied to solve an approximate system of linear equations arising from the Newton linearization. A variety of three-dimensional test cases have been conducted to demonstrate that this developed IRDG (P_1P_2) method is able to achieve a speedup of at least two orders of magnitude faster than its explicit counterpart, provided that a well approximated Jacobian matrix is necessarily implemented as the preconditioning matrix. The numerical experiments also indicate that this IRDG (P_1P_2) method can maintain linear stability in smooth flow and nonlinear stability at the presence of weak discontinuities, and deliver the desired third order of accuracy: an order of accuracy higher than that of the underlying DG (P_1) method, without significant increase in computing cost and memory requirements. The current development is focused on the extension of the IRDG method on tetrahedral grids for all speeds.

Acknowledgements

This manuscript has been authored by Battelle Energy Alliance, LLC under Contract No. DE-AC07-05ID14517 (INL/CON-09-16255) with the U.S. Department of Energy. The United States Government retains and the published, by accepting the article for publication, acknowledges that the United States Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The authors would like to acknowledge the partial support for this work provided by DOE under Nuclear Engineering University Program.

References

[1] Reed, Reed WH, Hill TR. Triangular mesh methods for the neutron transport equation. Los Alamos Scientific Laboratory Report, LA-UR-73-479, 1973.
[2] Cockburn B, Hou S, Shu CW. TVD Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case. Math Comput 1990;55:545–81.
[3] Cockburn B, Shu CW. The Runge–Kutta discontinuous Galerkin method for conservation laws V: Multidimensional system. J Comput Phys 1998;141:199–224.
[4] Cockburn B, Karniadakis G, Shu CW. The development of discontinuous Galerkin method, in discontinuous Galerkin methods, theory, computation, and applications. In: Cockburn B, Karniadakis GE, Shu CW, editors. Lecture notes in computational science and engineering, vol. 11. New York: Springer-Verlag; 2000. p. 5–50.
[5] Bassi F, Rebay S. High-order accurate discontinuous finite element solution of the 2D Euler equations. J Comput Phys 1997;138:251–85.
[6] Atkins HL, Shu CW. Quadrature free implementation of discontinuous Galerkin method for hyperbolic equations. AIAA J 1998;36(5).
[7] Bassi F, Rebay S. GMRES discontinuous Galerkin solution of the compressible Navier–Stokes equations, discontinuous Galerkin methods, theory, computation, and applications. In: Cockburn B, Karniadakis GE, Shu CW, editors. Lecture notes in computational science and engineering, vol. 11. New York: Springer-Verlag; 2000. p. 197–208.
[8] Warburton TC, Karniadakis GE. A discontinuous Galerkin method for the viscous MHD equations. J Comput Phys 1999;152:608–41.
[9] Hesthaven JS, Warburton T. Nodal discontinuous Galerkin methods: algorithms, analysis, and applications. Texts Appl Math 2008;56.
[10] Rasetarinera P, Hussaini MY. An efficient implicit discontinuous spectral Galerkin method. J Comput Phys 2001;172:718–38.
[11] Helenbrook BT, Mavriplis D, Atkins HL. Analysis of p-multigrid for continuous and discontinuous finite element discretizations. AIAA Paper 2003:2003–3989.
[12] Fidkowski KJ, Oliver TA, Lu J, Darmofal DL. p-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. J Comput Phys 2005;207(1):92–113.
[13] Luo H, Baum JD, Löhner R. A discontinuous Galerkin method using Taylor basis for compressible flows on arbitrary grids. J Comput Phys 2008;227(20):8875–93.
[14] Luo H, Baum JD, Löhner R. On the computation of steady-state compressible flows using a discontinuous Galerkin method. Int J Numer Meth Eng 2008;73(5):597–623.
[15] Luo H, Baum JD, Löhner R. A Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids. J Comput Phys 2007;225(1):686–713.
[16] Luo H, Baum JD, Löhner R. A p-multigrid discontinuous Galerkin method for the Euler equations on unstructured grids. J Comput Phys 2006;211(2):767–83.
[17] Luo H, Baum JD, Löhner R. A fast, p-multigrid discontinuous Galerkin method for compressible flows at all speed. AIAA J 2008;46(3):635–52.
[18] Dumbser M, Balsara DS, Toro EF, Munz CD. A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes. J Comput Phys 2008;227:8209–53.
[19] Dumbser M, Zanotti O. Very high order P_nP_m schemes on unstructured meshes for the resistive relativistic MHD equations. J Comput Phys 2009;228:6991–7006.
[20] Dumbser M. Arbitrary high order P_nP_m schemes on unstructured meshes for the compressible Navier–Stokes equations. J Comput Fluids 2010;39:60–76.
[21] Bassi F, Rebay S. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. J Comput Phys 1997;131:267–79.
[22] Bassi F, Rebay S. Discontinuous Galerkin solution of the reynolds-averaged Navier–Stokes and k-ω turbulence model equations. J Comput Phys 2005;34:507–40.
[23] Cockburn B, Shu CW. The local discontinuous Galerkin method for time-dependent convection–diffusion system. SIAM J Numer Anal 2001;16.
[24] Baumann CE, Oden JT. A discontinuous hp finite element method for the Euler and Navier–Stokes equations. Int J Numer Meth Fl 1999;31.
[25] Peraire J, Persson PO. The compact discontinuous Galerkin method for elliptic problems. SIAM J Sci Comput 2008;30:1806–24.
[26] H. Luo, L. Luo, R. Nourgaliev, V. Mousseau, A reconstructed discontinuous Galerkin method for the compressible Euler equations on arbitrary grids. AIAA-2009-3788, 2009.
[27] Luo H, Luo L, Nourgaliev R, Mousseau A, Dinh N. A reconstructed discontinuous Galerkin method for the compressible Navier–Stokes equations on arbitrary grids. J Comput Phys 2010;229:6961–78.
[28] Luo H, Luo L, Ali A, Nourgaliev R, Cai C. A parallel, reconstructed discontinuous Galerkin method for the compressible flows on arbitrary grids. Commun Comput Phys 2011;9(2):363–89.
[29] Haider DF, Croisille JP, Courbet B. Stability analysis of the cell centered finite-volume MUSCL method on unstructured grids. Numir Math 2009;113(4):555–600.
[30] Balsara D, Altmann C, Munz CD, Dumbser M. A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG + HWENO schemes. J Comput Phys 2007;226:586–620.
[31] Stoufflet B. Implicit finite element methods for the Euler equations. In: Angrand F, editor. Numerical methods for the Euler equations of fluid dynamics. Philadelphia: SIAM; 1985.
[32] Batina JT. Implicit flux-split Euler schemes for unsteady aerodynamic analysis involving unstructured dynamic meshes. AIAA J 1991;29(11).
[33] Venkatakrishnan V, Mavriplis DJ. Implicit solvers for unstructured meshes. J Comput Phys 1993;105(83).
[34] Knight DD. A fully implicit Navier–Stokes algorithm using an unstructured grid and flux difference splitting. AIAA Paper 1993:93–0875.
[35] Whitaker DL. Three-dimensional unstructured grid Euler computations using a fully-implicit, upwind method. AIAA Paper 1993:93–3337.
[36] Luo H, Baum JD, Löhner R, Fast A. Matrix-free implicit method for compressible flows on unstructured grids. J Comput Phys 1998;146(2):664–90.
[37] Barth TJ, Linton SW. An unstructured mesh Newton solver for compressible fluid flow and its parallel implementation. AIAA Paper 1995:95–0221.
[38] Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comp 1998;7(3):89.
[39] Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified analysis of discontinuous Galerkin methods for elliptic problems. SIAM J Numer Anal 2002;39(5.):1749–79.
[40] Gassner G, Lorcher F, Munz CD. A contribution to the construction of diffusion fluxes for finite volume and discontinuous Galerkin schemes. J Comput Phys 2007;224(2):1049–63.
[41] Liu H, Xu K. A Runge–Kutta discontinuous Galerkin method for viscous flow equations. J Comput Phys 2007;224(2):1223–42.
[42] Luo H, Xia Y. A class of reconstructed discontinuous Galerkin methods in computational fluid dynamics. In: International conference on mathematics

and computational methods applied to nuclear science and engineering, Brazil; May 2011.
[43] Luo H, Luo L, Nourgaliev R, Mousseau V. A reconstructed discontinuous Galerkin method for the compressible Navier–Stokes equations on arbitrary grids. AIAA-2010-0364, 2010.
[44] Luo H, Luo L, Norgaliev R, Mousseau VA, Dinh N. A reconstructed discontinuous Galerkin method for the compressible Navier–Stokes equations on arbitrary grids. J Comput Phys 2010;229:6961–78.
[45] Luo H, Luo L, Ali A, Norgaliev R, Cai C, Parallel A. Reconstructed discontinuous Galerkin method for the compressible flows on arbitrary grids. Commun Comput Phys 2011;9(2):363–89.
[46] P Zhang L, Liu W, He LX, Deng XG, Zhang HX. A class of hybrid DG/FV methods for conservation laws II: Two dimensional cases. J Comput Phys 2011. http://dx.doi.org/10.1016/j.jcp.2011.03.032.
[47] Batten P, Leschziner MA, Goldberg UC. Average-state Jacobians and implicit methods for compressible viscous and turbulent flows. J Comput Phys 1997;137:38–78.
[48] Roe PL. Approximate Riemann solvers, parameter vectors and difference schemes. J Comput Phys 1981;43(2):357–72.
[49] Pernice M, Walker HF. NITSOL: a Newton iterative solver for nonlinear systems. SIAM J Sci Stat Comput 1998;19:302–18.
[50] Illeim ER. CFD wing/pylon/finned store mutual interference wind tunnel experiment. AEDC-TSR-91-P4, Arnold (AFB, TN): Arnold Engineering Development Center; January 1991.